

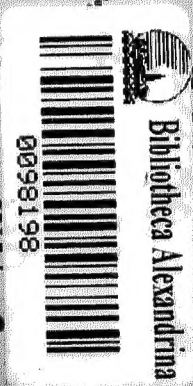
مهندس / (الرسمة الشبيبة)

تحدث إلى الكمبيوتر بلغة لوجو

لغة أصدقاء الروبوت



- اللغة التي نجعل من الكمبيوتر صديقاً لك
- لغة التحكم في الروبوت (الإنسان الآلي)
- لغة الذكاء الصناعي للكمبيوتر
- تطبيقات مختلفة على الرسم والألعاب





مكتبة ابن سينا

سلسلة علمية ثقافية نتناول مختلف العلوم والفنون ..

تصدرها

مكتبة القرآن

ويشرف عليها

مهندس / مصطفى عاصم

جميع الحقوق محفوظة
لمكتبة القرآن

مهندس / الأستاذ حسين

تحدث إلى الكمبيوتر

LOGO

بلغة لوجو

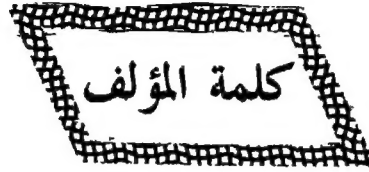
لغة أصدقاء الروبوت

LOGO

- اللغة التي تجعل من الكمبيوتر صديقاً لك .
- لغة التحكم في الروبوت (الإنسان الآلي) .
- لغة الذكاء الصناعي للكمبيوتر .
- تطبيقات مختلفة على الرسم والألعاب .

مكتبة القراء

للطبع والنشر والتوزيع
٣ شارع القماش بالفرنساوى - بولاق
القاهرة - ت ٧٦١٩٦٢ - ٧٦٨٥٩١



● ما هي لوجو ؟

لغة « لوجو » لغة مختلفة تماماً ..

تختلف عن سائر لغات الكمبيوتر في الشكل والبناء . وتتميز بخصائصها الفريدة في الرسم علاوة على كونها لغة « صديقة » ألفاظها تشبه ألفاظ اللغة العادية التي يتحدثها الناس « بالإنجليزية طبعاً » ! . والاسم « لوجو » ليس اختصاراً لعبارة ما كما هو معروف عن لغات الكمبيوتر الأخرى ولكنه مشتق من كلمة يونانية تعنى « الكلمة » أو « الفكرة » !

وقد قام بتصميم لغة لوجو فريق من الباحثين وقام بتطويرها إلى صورتها النهائية التي هي بها الآن الباحث « سيمور بابرت » (S.Papert) في معامل الذكاء الصناعي بولاية « ماساتشوستس » .

ولغة لوجو لها ارتباط وثيق بلغة LISP التي صممت خصيصاً لأبحاث الذكاء الصناعي فهي تعتبر تطويراً للغة LISP بغرض تسهيل مهمة الأطفال في تعلم الكمبيوتر وبرمجته في سن مبكرة جداً ..

لذلك تعتبر لغة لوجو هي اللغة المناسبة لجذب الأطفال إلى عالم الكمبيوتر بدءاً من سن المدرسة . ومع ذلك فسهولة اللغة لا تمنع من كونها لغة قوية تفيد الجميع صغاراً وكباراً . وسهولة اللغة لا تعنى أكثر من أن جهداً كبيراً قد بذله واضعوا اللغة حتى يجعلوا منها لغة سهلة .

ومن الاستخدامات الهامة للغة الرسم بكفاءة عالية ، وخدمة أغراض الذكاء الصناعي والتحكم في الروبوت فضلاً عن أغراض البرمجة العامة .

وتنفرد لغة لوجو باحتوائها على منشآت للتحكم (control structures) التي تجعلها قادرة على معالجة البيانات في قوائم (lists) . كما أنها تتميز بالبرامج التي تستدعي نفسها بنفسها وهي خاصية لا تتوفر لكل اللغات الأخرى .

● طرازات لوجو :

توفرت لغة لوجو للعديد من أجهزة الكمبيوتر نذكر منها :

- | | |
|---------------|------------------------------|
| ١ — الكمبيوتر | apple II |
| ٢ — الكمبيوتر | IBM والأجهزة المتوافقة معه . |
| ٣ — الكمبيوتر | Radioshake |
| ٤ — الكمبيوتر | كومودور ٦٤ |
| ٥ — الكمبيوتر | تكساس المنزلي TI-99/4A |
| ٦ — الكمبيوتر | سنكلير Zxspectrum |
| ٧ — الكمبيوتر | صخر MSX |

وقد قامت شركات البرامج بتطبيق لغة لوجو الآن على معظم أجهزة الكمبيوتر . ولا توجد فوارق كبيرة بين اللهجات المختلفة للغة ، وهي تنحصر أساساً في استخدام أجهزة الطباعة والقرص المغنطيسي أو الكاسيت ، لذلك فهي لغة قياسية . وبحسب الجهاز يمكنك أن تحصل على لغة لوجو مسجلة على شريط كاسيت أو كارتريج أو على قرص مغنطيسي .

أقدم هذا الكتاب عن البرمجة بلغة لوجو آملاً أن يكون إضافة حقيقية لمكتبة الكمبيوتر العربية وأن يستمتع به القراء صغاراً وكباراً ..

والله ولي التوفيق ..

أسامة الحسيني

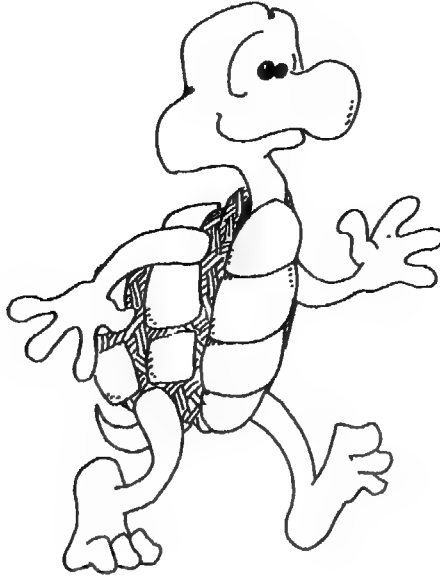
۱

لنكتشف معاً عالم لوجو

● لننتعرف بعائلة لوجو :

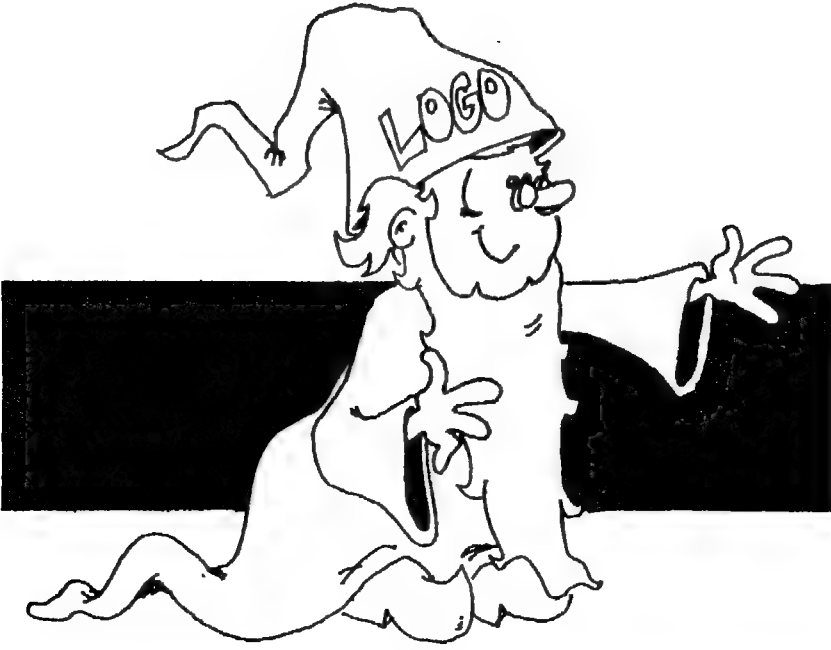
إن لغة لوجو لغة حيّة مثيرة . وسوف نتحدث عنها من خلال بعض الأصدقاء الذين يساعدوننا على شرح بعض جوانب اللغة . وبعض الأصدقاء موجود باللغة نفسها مثل السلحفاة البحرية (Turtle) التي تساعدنا في الرسم بالكمبيوتر ، أما الشخصيات الأخرى فسوف تصاحبنا فقط في هذا الكتاب فلننتعرف بهم الآن :

١ السلحفاة :



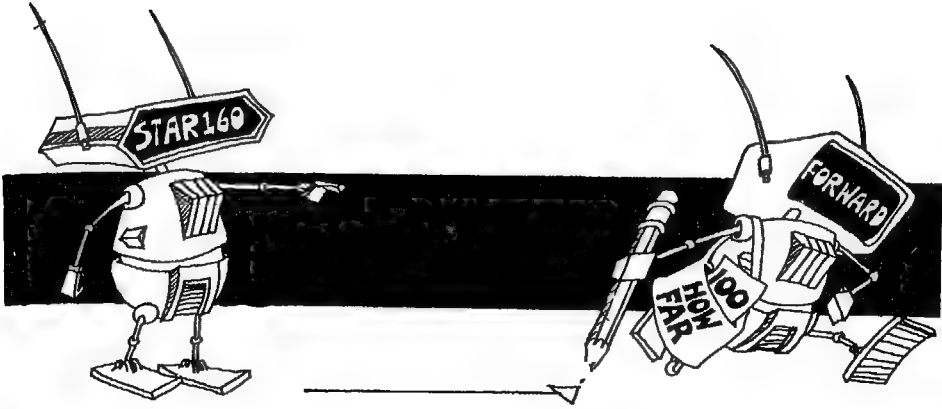
هذه هي الشخصية الرئيسية في لغة لوجو وسوف تتعامل معها كثيراً عند الرسم على الشاشة ، فهي التي تتلقى منك الأوامر وتقوم بتنفيذها أمامك خطوة بخطوة . ومع بعض أجهزة الكمبيوتر يمكن أن تكون السلحفاة عبارة عن روبوط حقيقي .

٢ □ الساحر لوجو :



الساحر « لوجو » يختفي بداخل الكمبيوتر ولن نراه على الشاشة للأسف . فهو مشغول طوال الوقت لأنه مسئول عن تلقي جميع الأوامر ومتابعة تنفيذها بواسطة الروبوتات (Robots) التي تساعد في تنفيذ الأوامر . وسوف نلتقي بالساحر لوجو أحياناً على صفحات الكتاب ليشرح لنا بعض أسرار عمله « عما يحدث بداخل الكمبيوتر » ...

٣ الروبوتات المساعدة :

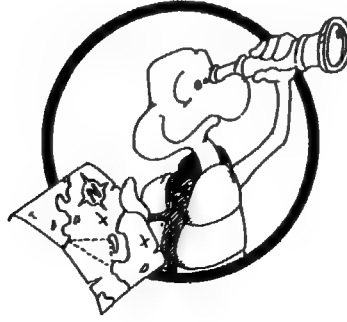


أما الروبوتات المساعدة .. فهي تمثل أوامر لغة لوجو التي يقوم كل منها بأداء عمل محدد يختص به . وهناك نوعان من الروبوتات النوع الأول ذو الشاشة المربعة مثل الروبوت FORWARD الموضح بالرسم وهو يمثل أمراً من أوامر اللغة المعروفة .. أما الروبوت الآخر ذو الشاشة المسدسة STAR160 فهو روبوت مبتكر ! وهو يمثل الأوامر الجديدة التي يمكنك أن تعلمها للكمبيوتر كما سنعرف عند تقدمنا في أبواب الكتاب .

وسواء كانت الروبوتات أصلية أو مبتكرة ، فسوف نلتقي معها عندما يقوم الساحر بشرح « ما يحدث بداخل الكمبيوتر » .

٤ أصدقاء لوجو :

هؤلاء الأصدقاء سوف يتجولون معنا في رحلتنا داخل هذا الكتاب . أما الصديق الأول فهو يمّسك بنظارة معظّمة ويظهر لنا عند التمارين والتجارب التي نقدمها والتي تتطلب منك قدراً من التفكير أو الاستكشاف باستخدام الكمبيوتر .



الصديق الثاني ... هما اثنان متلازمان في الحقيقة أحدهما يمّسك بكشاف لينير الطريق أمامنا . ويظهر هذان الصديقان عند النقاط التي يجب علينا قراءتها بعناية بالغة حيث تكون نقاطاً حاكمة أو ملاحظات هامة .



أما الصديق الثالث فهو يصادفنا عندما نلتقى بفكرة جديدة تتطلب البحث والتقصّي من الأصدقاء هواة البحث والتقصّي .



وبصفة عامة فإن التجارب والتمارين التي يعز عليك تنفيذها سوف تجدها محلولة إما في آخر الكتاب أو مع موضوعات الكتاب نفسه عندما تتقدم في القراءة فصلاً بعد آخر .

● إعداد الكمبيوتر للعمل :

أياً كان الكمبيوتر الذى تعمل عليه فإنك تحتاج دائماً إلى « مترجم لغة لوجو » الذى سيجعل الكمبيوتر يفهم أوامر لوجو التى يتلقاها . معنى هذا أنك تشتري لغة « لوجو » منفصلة عن الكمبيوتر ، بخلاف « بيسك » . فكل أجهزة الكمبيوتر الصغيرة تستطيع أن تفهم لغة « بيسك » من تلقاء نفسها . وبمجرد أن تضغط على زر التشغيل يمكنك أن تبدأ فى إعطاء الأوامر بلغة بيسك .

أما لغة لوجو فهى عادة تكون مخزنة على شريط كاسيت أو قرص مغنطيسى أو « كارتريديج » .

فإذا كانت اللغة على قرص أو كارتريديج فالأمر لا يتطلب منك سوى وضع القرص أو الكارتريديج فى المكان المخصص له فى الكمبيوتر والضغط على زر التشغيل فيبدأ الكمبيوتر التعامل بلغة لوجو .

● فالحهاز تكساس TI-99/4A مثلاً يحقق لك استخدام لغة لوجو من خلال كارتريديج TI LOGO II .

● كذلك الجهاز صخر MSX يملك بلغة لوجو على الكارتريديج « صخر لوجو » الذى يجعل الكمبيوتر يتحدث لوجو باللغتين العربية والإنجليزية .

● أما مع الكمبيوتر سنكلير (Zx spectrum) فيمكنك شراء لغة لوجو مسجلة على شريط كاسيت .

● ومع الجهاز IBM والأجهزة المماثلة له فعادة تكون لغة لوجو مخزنة على قرص مغنطيسى (علاوة على بعض الإمكانيات الإضافية على قرص منفصل هو (LWIL) .

وفى حالة ما إذا كانت اللغة مسجلة على شريط كاسيت فإنك تقوم بنقلها (تحميلها) إلى ذاكرة الكمبيوتر تماماً مثل برامج الألعاب العادية ؛ باستخدام الأمر LOAD غالباً . وهذا لا يمنع نقلها إلى القرص المغنطيسى لو توفّر لديك .

وفى جميع الأحوال عندما يبدأ الكمبيوتر العمل بلغة لوجو سوف تشاهد رسالة ترحيب على الشاشة مثل :

WELCOME TO LOGO

عندئذ يمكنك أن تبدأ .

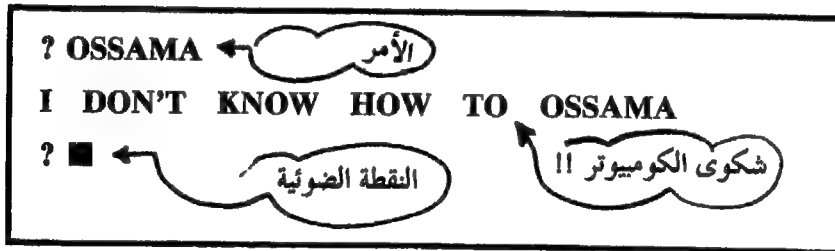
● لنكتب أمراً بلغة لوجو ...

عندما يكون الكمبيوتر مستعداً لتلقى أوامر لوجو سوف تشاهد علامة الاستعداد (؟) على يسار الجزء العلوى من الشاشة يليها النقطة الضوئية (cursor) .

ولتبدأ بكتابة اسمك ولنشاهد معاً ما يحدث :

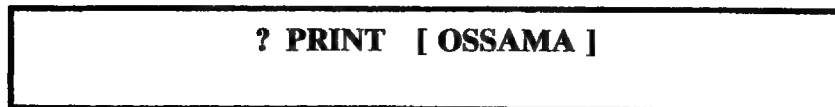


بعد الانتهاء من كتابة الاسم "OSSAMA" سوف تكون النقطة الضوئية تخفق على يمين الاسم . فإذا ضغطت على الزر (ENTER) فإن الأمر المكتوب يدخل إلى الكمبيوتر وترى الرسالة الآتية على الشاشة :

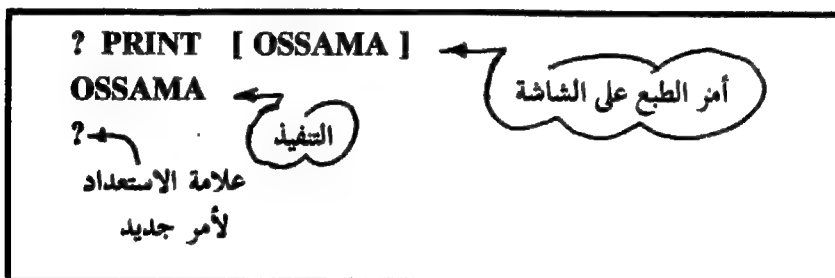


إن الكمبيوتر يشتكى لأنه لا يعرف كيف ينفذ هذا الأمر الغريب .. لأن كل ما يكتب على يمين علامة الاستعداد (?) في الحقيقة يمثل أمراً للكمبيوتر . (COMMAND)

فإذا أردت إعطائه أمراً لكتابة اسمك فلنستخدم الأمر PRINT بالصورة الآتية :



عندئذ يظهر الاسم على الشاشة مطبوعاً تحت الأمر مباشرة كالآتي :



إذن فقد تعلمنا أول أمر بلغة لوجو وهو أمر الطباعة PRINT ولا ننسى استخدام الأقواس المربعة [] لنضع بينهما ما نريد طباعته على الشاشة .

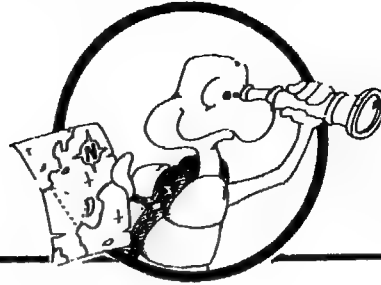
تجربة

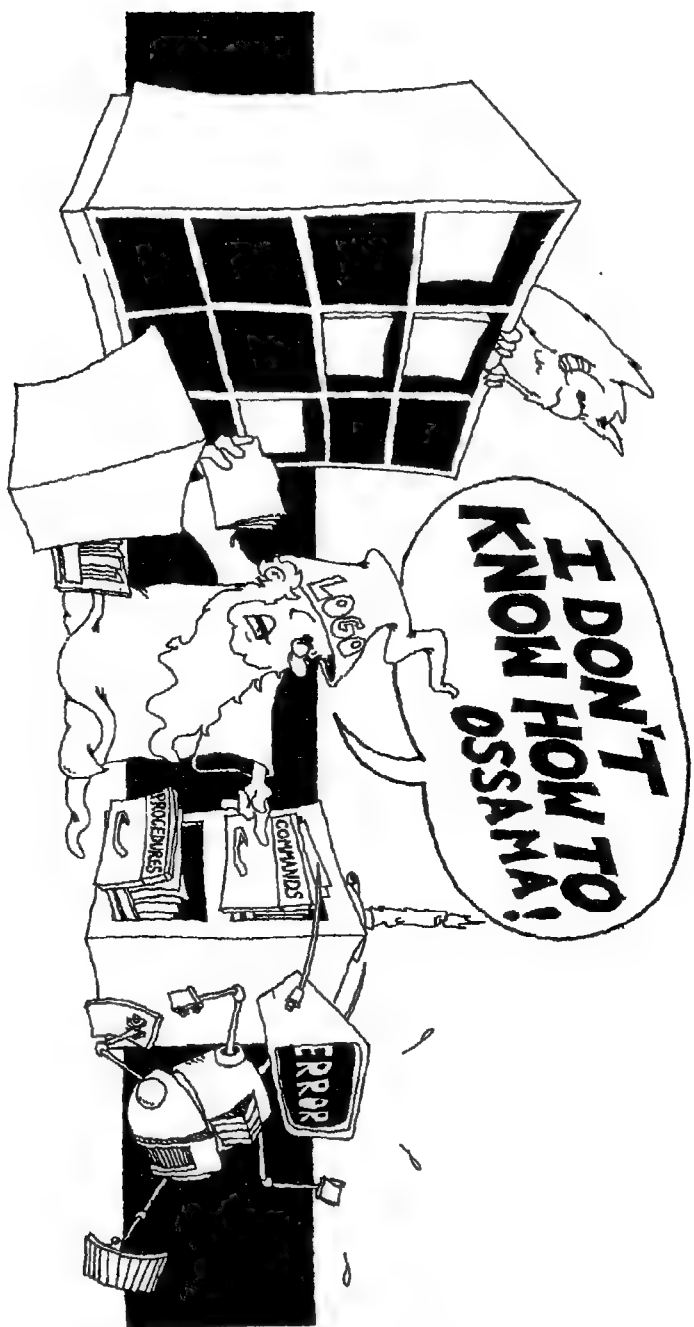
* عرفنا الآن أن إعطاء الأمر للكمبيوتر بلغة لوجو له صيغة معينة .
فإذا خرجنا عنها فإنه يشكو ، وعادة نسمى الشكوى « رسالة خطأ »
(error message) .

والكمبيوتر كثير الشكوى عندما لا يفهم ما نريده أن يفعله .
فلتجرب الآن بعض هذه الأوامر ذات الصور والأشكال المتعددة ، لكي
تعرف أى الأوامر ستجعله يشكو وأيها سوف ينفذه .

وهذا تدريب جيد لكي تتعرف على بعض رسائل الأخطاء التي
يوجهها إليك الكمبيوتر بلغة لوجو :

PRINT
PRINT OSSAMA
PRINTOSSAMA
PRINT [OSSAMA]





الساحر لوجو .. يشكو من عدم فهمه للأمر !

فلاش

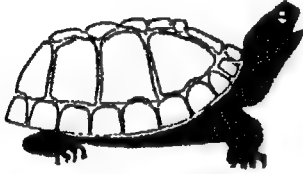


هل لاحظت أننا نستخدم الحروف الكبيرة في لغة لوجو ؟ .. إنها قاعدة قياسية أن نكتب لغة لوجو بالحروف الكبيرة ومع ذلك فسوف نصادف بعض الأوامر مكتوبة بالحروف الصغيرة فبعض أجهزة الكمبيوتر تتسامح في هذه النقطة . لكنها عادة حميدة أن تلتزم بالقاعدة حتى تتقبل أوامرك كل أجهزة الكمبيوتر .

● لا تخش أن تتلف الكمبيوتر :

إنه شيء مفيد حقاً عندما تشرع في تعلم لغة مثل لوجو — وهي تتميز بالتنفيذ الفوري للأوامر المعطاه لها — أن تجرب إعطاء ما تشاء من الأوامر وترتكب ما شئت من الأخطاء . فبذلك يمكنك أن تتعرف على حدود الكمبيوتر في لغة لوجو وتتعرف على الأخطاء التي يتقبلها والأخطاء التي يشكو منها .

ولا تخش أن تعطى الكمبيوتر أمراً ما . فلن يتلف من الأوامر . وجرب واكتشف .



● هيا نلتقي بالسلحفاة :

والآن نحن على استعداد لمقابلة السلحفاة التي سوف تساعدنا على الرسم بالكمبيوتر .

ولنكتب الآن الأمر الآتي ثم نضغط على الزر ENTER :

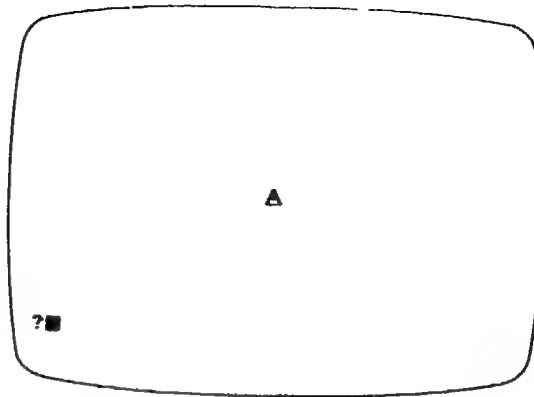
CS

هل شاهدت النتيجة ؟

إنه أمر تنظيف الشاشة وهو اختصار للكلمة الآتية :

CEEARSCREEN

ويمكنك استخدام الكلمة كاملة أو الاختصار CS أيهما تحب . ففي الحالتين سوف تحصل على شاشة نظيفة تماماً من الكتابة السابقة وسوف ترى السلاحف البحرية في وسط الشاشة وعلامة الاستعداد تنتظرك في الركن الأيسر السفلي هذه المرة .



السلاحف في منتصف الشاشة بعد تنفيذ الأمر CS

والسلاحف كما ذكرنا يمكن استبدالها ببروابط حقيقي مع بعض طرازات الكمبيوتر . لكنها مع ذلك وهي على شاشة الكمبيوتر يمكن توجيهها لتحرك

في أى اتجاه باستخدام أوامر مفهومة واضحة المعنى مثل « إلى الأمام .. عشر خطوات » أو « إلى الخلف خمس خطوات » أو « استدر اليمين أو اليسار » .
وبهذا فإن السلحفاه بالنسبة لمبرمج لوجو هي هذا الرّحّال الذى يحمل خريطة عليها إرشادات تدله على الطريق وسط الصحراء القاحلة .



السلحفاه تحمل الخريطة كمرشد للطريق في الصحراء

اكتب الآن الأمر الآتى ثم اضغط على الزر ENTER (الزر ذو العلامة ↓) في الجهاز (IBM) :

المعنى : إلى الأمام ٥٠ خطوة ← **FORWARD 50 ?**

ماذا حدث على الشاشة ؟

لقد تحركت السلحفاه إلى الأمام بمقدار ٥٠ خطوة .

أعط الآن الأمر الآتى يليه الضغط على الزر ENTER :

المعنى : إلى الخلف ١٠٠ خطوة ← **BACK 100 ?**

تتحرك السلحفاه للخلف مائة خطوة حتى تصبح في الجزء السفلى من الشاشة .

الآن يمكننا تحريك هذه السلحفاه في الاتجاهات الأربعة باستخدام أربعة أوامر هي :

الأمر	اختصار الأمر	المعنى
FORWARD	FD	للأمام
BACK	BK	للخلف
LEFT	LT	لليسار
RIGHT	RT	لليمين

وسوف نلاحظ دائماً أن لغة لوجو تتميز بأنها تمنحنا الاختيار بين استخدام كلمات ذات معنى مثل LEFT و RIGHT أو كلمات مختصرة مثل LT ، RT مكافئة لها تماماً . وتتميز الكلمات الكاملة بأنها لا يمكن أن تُنسى لا سيما إذا كنّا نعرف معناها أما الاختصارات فهي توفر الوقت .

وهذه الأوامر الأربعة تحتاج معها لبعض الأرقام حتى يكتمل معنى الأمر . فلنجرب هذه المجموعة من لأوامر المتتالية :

اكتب هذه الأوامر واحداً بعد الآخر وأدخل كل أمر بالضغط على الزر

: ENTER

FORWARD 40
RIGHT 30
FORWARD 50
BACK 80
LEFT 90

مسافة خالية

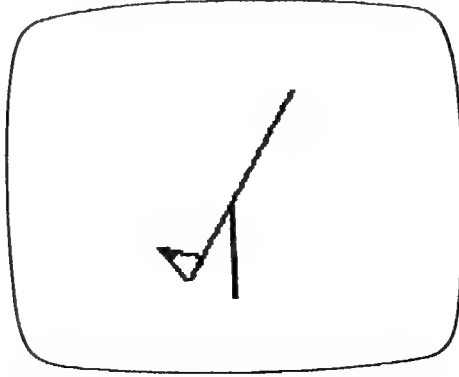
أو أدخل الأوامر مختصرة بهذه الصورة :

FD 40
RT 30
FD 50
BK 80
LT 90

مسافة غالية

ماذا حصلت عليه بعد تنفيذ هذه الأوامر ؟

لو أنك كتبت هذه الأوامر بصيغتها السليمة (سواء العادية أو المختصرة) فإنك سوف تحصل في النهاية على الشكل الآتي :



نتيجة تنفيذ مجموعة أوامر الحركة للسلاحف

فلنتفهم معاً نتيجة كل أمر على حدة :

الأمر الأول FD 40 معناه « تحرك إلى الأمام بمقدار ٤٠ خطوة » .



FORWARD 40

الأمر الثاني 30 RT معناه « استدر إلى اليمين بزاوية قدرها ٣٠ درجة » .



RIGHT 30

الأمر الثالث 50 FD معناه « تحرك إلى الأمام بمقدار ٥٠ خطوة » ..
وبالطبع ستكون الحركة في الاتجاه الجديد الذي يحدده السهم .



FORWARD-50

الأمر الرابع 80 BK معناه « تحرك عكس اتجاه السهم أو إلى الخلف بمقدار ٨٠ خطوة » .



BACK 80

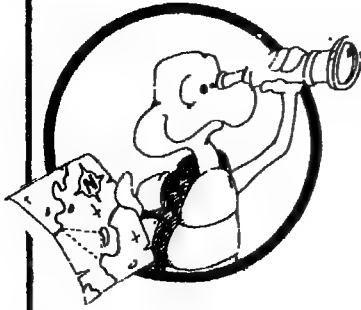
أما الأخير فمعناه تغيير الاتجاه ولكن إلى اليسار (LEFT) بمقدار ٩٠ درجة ،
وينتج عنه الرسم الموضح بالشكل النهائي .

نخرج من هذا بأن الرقم الذى يعقب كل من FD ، BK يمثل عدد الخطوات التى تتحركها السلحفاه .

أما الأرقام التى تعقب كل من RT ، LT فهى تمثل زاوية دوران السهم يمينا أو يساراً .

تجربة

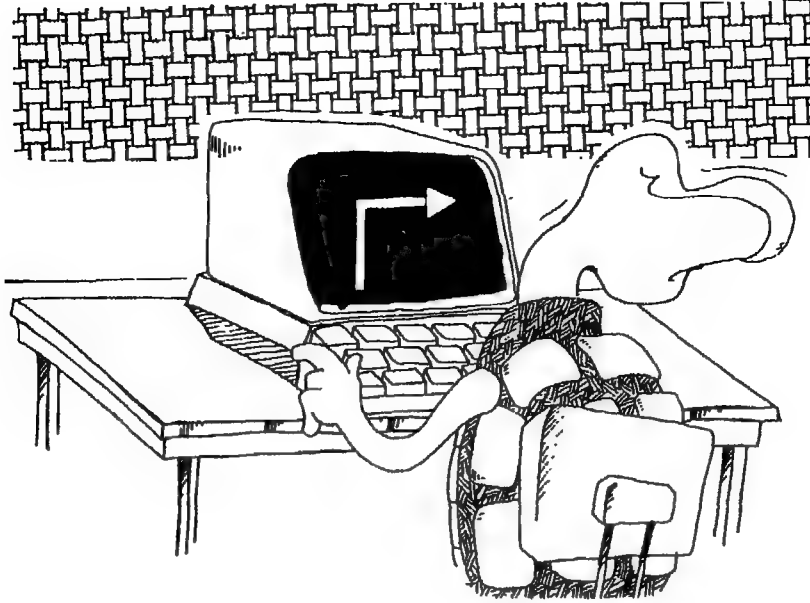
● لو أنك نسيت أن تكتب رقماً بعد أمر الحركة FD فماذا تتوقع أن تكون الإجابة من لغة لوجو ؟



● جرب أيضاً إعطاء أرقام مختلفة للحركة فى جميع الاتجاهات حتى يمكنك أن تعرف عدد الخطوات التى تتسع لها الشاشة كلها .. وبالطبع فإن كل كومبيوتر له اتساع مختلف لشاشته . جرب وسجل النتائج .

٢

التحكم في السلحفاه



• عالم السلحفاه :

إن عالم السلحفاه مازال متسعاً وغنياً بأوامر الحركة والرسم. والتحكم والتلوين .

وقبل أن نتعرف بالمزيد من الأوامر الجديدة دعنا نلخص ما تعلمناه أولاً :
* الأمر **CLEARSCREEN** أو **CS**

يسمح ما على الشاشة من كتابة أو رسومات ويضع السلحفاه في المنتصف تماماً جاهزة لتلقى أوامر جديدة .

* الأمر **FORWARD** أو **FD**

والأمر **BACK** أو **BK**

للحركة للأمام وللخلف ، وكلاهما يحتاج لإدخال رقم يمثل عدد الخطوات ونسميه الرقم المُدخل (INPUT) .

* الأمر **RIGHT** أو **RT**

والأمر **LEFT** أو **LT**

لدوران السلحفاه يميناً أو يساراً وتحتاج أيضاً هذه الأوامر لرقم مُدخل يمثل زاوية الدوران بالدرجات .

ملاحظة :

يجب ترك مسافة خالية بين الأمر وبين الرقم المُدخل معه مثل :

LEFT 45

LT 45 أو

مسافة خالية

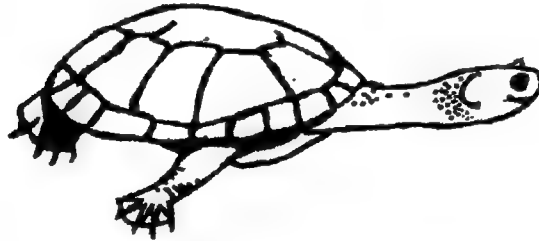
وبدون هذه المسافة سوف يشكو الكمبيوتر !



وبالإضافة إلى ما سبق ربما تحب أن تكتب أعلى الشاشة وفي هذه الحالة تستخدم الأمر الجديد TEXTSCREEN بمعنى شاشة الكتابة ويختصر إلى TS .

وفي هذه الحالة تختفي السلحفاه من منتصف الشاشة وترى علامة الاستعداد أعلى يسار الشاشة . وهذه الشاشة تصلح لأداء العمليات الحسابية أو لكتابة برنامج لوجو كما سنعرف في الأجزاء المقبلة .

ومع ذلك فعند إعطاء أى أمر من أوامر الحركة التي عرضناها فإن الكمبيوتر يعود تلقائياً إلى شاشة السلحفاه .



● المزيد من الأوامر للسلحفاه PENUP, PENDOWN :

قد نرغب أحياناً في رفع القلم عن صفحة الرسم ونقله إلى موضع ما وبدء الرسم من هذا الموضع الجديد .

وكل ما نقوله في لغة لوجو للسلحفاه هو الأمر « ارفع القلم » أو بالإنجليزية PENUP فإذا تلقت السلحفاه هذا الأمر فإنها تتحرك وفقاً للأوامر التي تتلقاها ولكنها لا ترسم لأن القلم مرفوع .

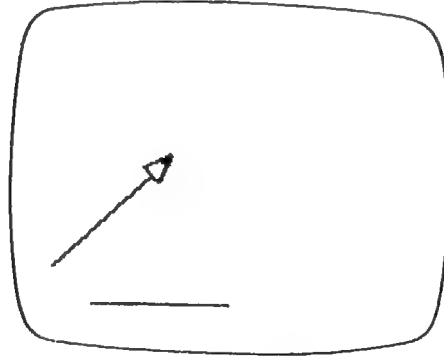
فإذا أردنا أن نبدأ في الرسم من جديد فإننا نعطيهما الأمر « اخفض القلم » أو PENDOWN عندئذ تبدأ السلحفاه في الرسم من جديد وفقاً للأوامر التي تتلقاها . والأمر PENUP يمكن اختصاره إلى PU . والأمر PENDOWN يمكن اختصاره إلى PD .

هل نبدأ في الرسم من جديد باستخدام هذين الأمرين ؟

لنعط الكومبيوتر أولاً الأمر CS لتنظيف الشاشة ثم تبدأ في تنفيذ مجموعة الأوامر التالية :

```
CLEARSCREEN
PENUP
FORWARD 50
RIGHT 90
PENDOWN
FORWARD 50
PENUP
RIGHT 45
FORWARD 20
PENDOWN
RIGHT 90
FORWARD 50
```

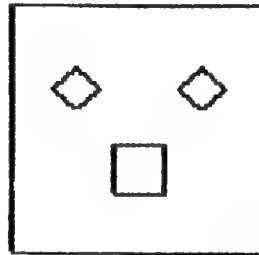
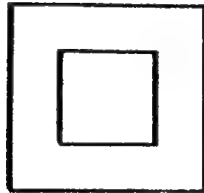
والشكل الآتي بعد يوضح نتيجة تنفيذ هذه المجموعة من الأوامر :

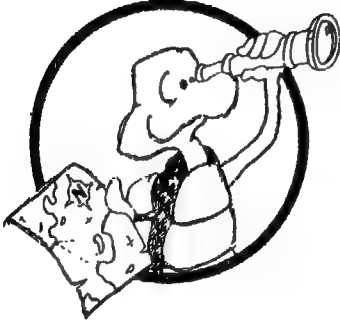


نتيجة تنفيذ مجموعة الأوامر السابقة باستخدام PU , PD

تمرين

هل تستطيع الآن أن ترسم الأشكال الموضحة بعد في الشكل باستخدام ما تعلمناه من أوامر .
لا بأس أن تحاول الآن فإذا لم توفق فسوف نعرف معاً كيفية تنفيذ مثل هذه الرسومات في الفقرات المتقدمة من الكتاب .





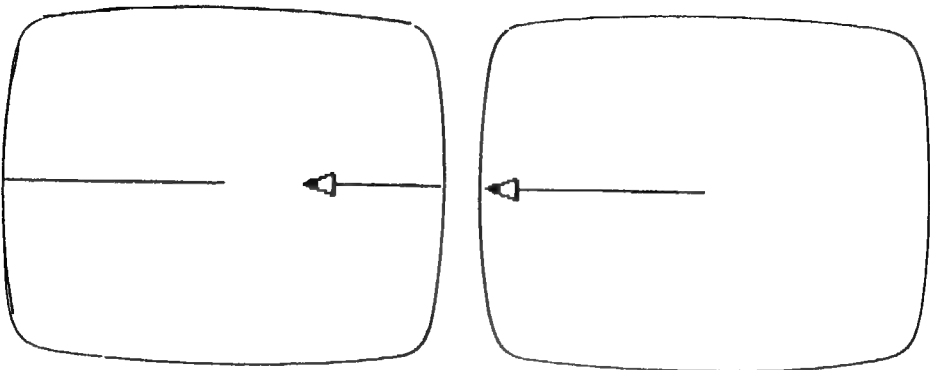
● السلاحفاه تُخزِم الشاشة :

بعد أن تتعرف على عدد الخطوات التى يمكن أن تحتويها الشاشة فى الاتجاهين الأفقى والرأسى . يمكنك أن تعرف إحداثيات أى نقطة على الشاشة . وعدد الخطوات يختلف من جهاز إلى آخر .

ولكن من التجارب الطريفة التى يمكنك أن تجربها إعطاء الأمر FD مع تجربة بعض الأرقام التى تعتقد أنها تعبّر عن بعد النقطة التى تقع أعلى الشاشة . ومرة بعد مرة سوف تعثر على الرقم المطلوب .

وكرر الأمر أيضاً بالنسبة ليمين ويسار وأركان الشاشة .

ومن الخصائص المميزة للسلاحفاه أنها لن تشكو إذا أعطيتها أرقاماً أكبر من حدود الشاشة فهى تعود دائماً من الجهة الأخرى كما فى الشكل التالى :

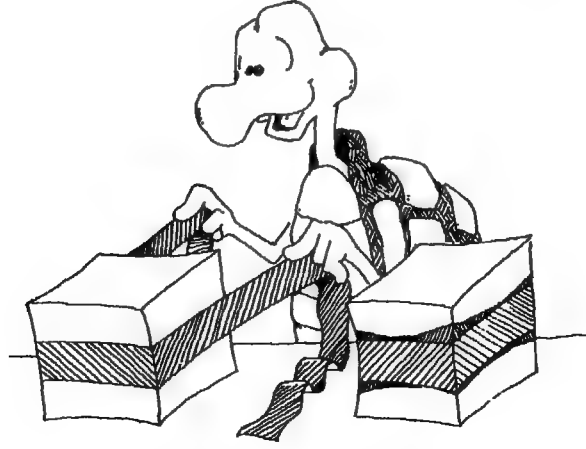


(ب) السلاحفاه تعود من الجهة الأخرى

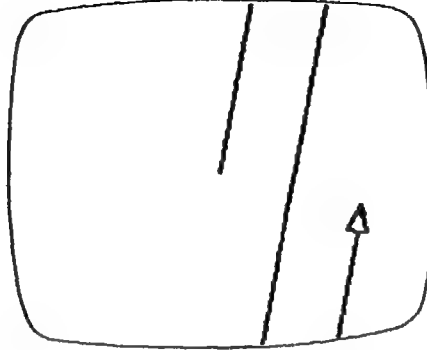
(أ) السلاحفاه فى أقصى اليمين

بل يمكن استخدام هذه الخاصية في تخزين الشاشة تماماً كما تقوم بتخزين علبة هدية بشرط من « السوليقان » .

وهذا يتم بإدارة السلحفاه قليلاً وإعطاء الأمر FD مع رقم كبير جداً .



السلحفاه تقوم بتخزين الهدايا



خاصية التخزين wrapping

● لنرسم مربعاً بالسلحفاة :
أعط سلسلة الأوامر التالية :

FD 50

RT 90

FD 50

إن هذه الأوامر ترسم جزءاً من مربع ، فالأمر الأول يقول « إلى الأمام خمسين خطوة » ، والثاني يقول « استدر جهة اليمين ٩٠ » ، والثالث يقول « إلى الأمام خمسين خطوة » . بذلك نحصل على الأشكال التالية واحداً بعد الآخر .



نتائج خطوات تنفيذ أوامر رسم المربع

هل تستطيع إكمال الشكل بحيث تُكوّن مربعاً كاملاً ؟

إن كل المطلوب هو إدارة السلحفاة بعد ذلك إلى أسفل وتحريكها نفس عدد الخطوات ثم إدارتها إلى اليسار وإعادتها إلى الموضع الأول بنفس عدد الخطوات .

FD 50

وهذه هي التعليمات كاملة :

RT 90 ← إلى اليمين .

FD 50

RT 90 ← إلى أسفل

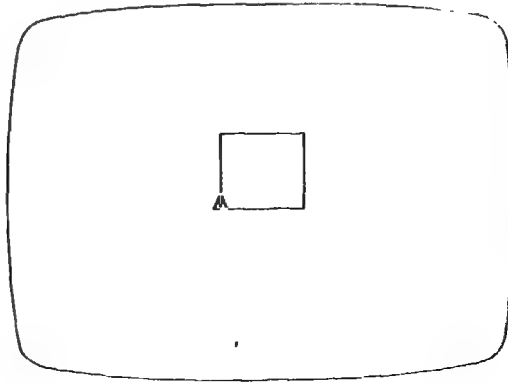
FD 50

RT 90 ← إلى اليسار

FD 50

RT 90 ← إلى أعلى

المربع الناتج والسلفاء
في وضعها الأصلي



ملاحظة



قد يبدو المربع في بعض الأحيان
مستطيلاً قليلاً وهذا يعتمد على الشاشة
المستخدمة فلا تقلق . كما يحدث ذلك
أحياناً مع جهاز الطباعة وسوف نلاحظ
هذا مع بعض الأمثلة .:

تجربة



● إخفاء السلحفاة .. وإظهارها

HIDETURTLE,SHOWTURTLE

بقى شيء هام ينقص المربع الذي قمنا برسمه الآن ، ولا شك أنك لاحظته أيضاً عند محاولتك رسم المستطيلات . هذا هو السلحفاة نفسها التي تبقى دائماً معنا بعد انتهائنا من رسم المربع .

ولا شك أنك ترغب في إخفائها لمشاهدة المربع كاملاً .

يتم ذلك بإعطاء الأمر HT وهو اختصار الأمر :

HIDETURTLE

بمعنى « أخف السلحفاة » عندئذ تختفى السلحفاة من الرسم وترى المربع « نظيفاً » .

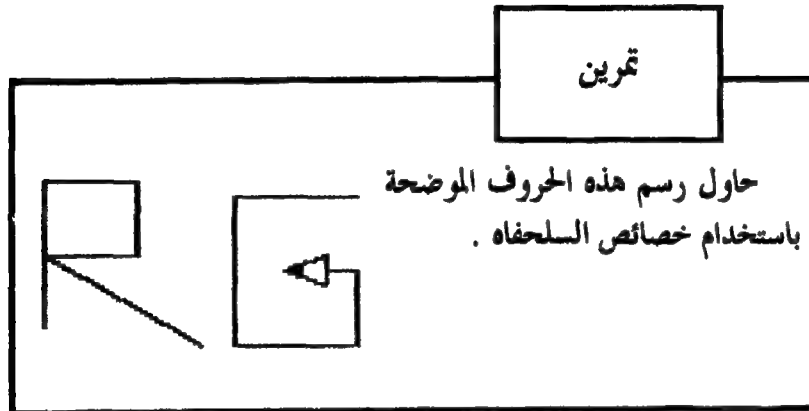
وبالطبع يمكن إعطاء الأمر بأحد صورتيه مختصراً أو كاملاً .

ولإعادة السلحفاة إلى الشاشة يعطى الأمر « أظهر السلحفاة » أى SHOWTURTLE أو بالصورة المختصرة ST . يمكنك الآن إضافة الأمر HT في نهاية مجموعة الأوامر السابقة وشاهد النتيجة .

وهذا الأمر الجديد سوف يكون نافعاً عند رسم المستطيلات والأشكال الأخرى التي تحاول رسمها .

وهذا ملخص بكلما لوجو التي تعلمناها مؤخراً :

المعنى	اختصار الأمر	الأمر
إعداد شاشة الكتابة بدون سلحفاة .	TS	TEXTSCREEN
ارفع القلم .. لا ترسم عند التحرك .	PU	PENUP
اخفض القلم .. ارسم عندما تتحرك .	PD	PENDOWN
أخف السلحفاة .	HT	HIDETURTLE
أظهر السلحفاة .	ST	SHOWTURTLE



SETPAL, SETPC

● الرسم بالألوان^(١)

يمكن تكوين الرسومات باستخدام أمرين جديدين هما :

اختيار مجموعة الألوان SETPAL

(وهو يعنى بالإنجليزية (set palette

٢ — اختيار لون القلم SETPC

(وهو يعنى بالإنجليزية (set pen color

وعادة يتبع الأمر SETPAL إما الرقم 0 أو الرقم 1 .

أما الأمر SETPC فيعقبه أحد الأرقام من 1 إلى 3 .

ومن هذه التوليفة يمكن الحصول على ستة ألوان مختلفة للرسم كالآتي :

● أعط أمر ألوان المجموعة الأولى ← STPAL 1

يمكن مع هذه المجموعة إعطاء الأوامر التالية لكل لون :

1 SETPC (اللون الأزرق الفاتح cyan)

2 SETPC (اللون الأرجواني magenta)

3 SETPC (اللون الأبيض white)

● أو أعط أمر اختيار ألوان المجموعة الثانية SETPAL 0

ويمكن مع هذه المجموعة إعطاء الأوامر التالية لكل لون :

1 SETPC (اللون الأخضر green)

2 SETPC (اللون الأحمر red)

3 SETPC (اللون البني brown)

(١) تختلف عبارات الألوان باختلاف الأجهزة .

تجربة

جرب الآن إعطاء أمر تلوين أحد المجموعتين (0) أو (1) يليه اختيار لون القلم الذى يروق لك ثم ارسم شكلاً ملوّناً .
(ويمكنك تغيير لون القلم عدة مرات أثناء الرسم) .

SETBG

● تلوين الخلفية

يوجد لون رابع خلاف الألوان 1 ، 2 ، 3 ، بكل مجموعة ألوان ، هذا هو اللون صفر (0) . وهذا اللون يرسم بنفس لون الخلفية ولذلك فهو يسمح الرسم الموجود على الشاشة إذا مرّ فوق .

علاوة على ذلك فإنه يمكن تغيير لون الخلفية باستخدام الأمر SETBG مع إدخال رقم بعدها من (0 إلى 15) وبذلك نحصل على ١٦ لوناً مختلفاً للخلفية .

● مثال :

الآتى بعد مجموعة من الأوامر لرسم مربع ذى أضلاع ملونة بالألوان الأخضر والأحمر والبنى والأخضر بالترتيب وذلك على خلفية بيضاء .

المجموعة 0
لون الخلفية 7 (أبيض)
SETPAL 0 ←
SETBG 7 ←
SETPC 1 ← أخضر
FORWARD 50
RIGHT 90
SETPC 2 ← أحمر
FORWARD 50
RIGHT 90
SETPC 3 ← بنى
FORWARD 50
RIGHT 90
SETPC 1 ← أخضر
FORWARD 50



● الألوان .. وأجهزة الكمبيوتر المختلفة :

نظراً لأن خاصية التلوين ترتبط دائماً بنوع الكمبيوتر ، حيث أنها تعتمد على تصميم المعدة نفسها ، لذلك فإننا يجب أن نشير إلى أن طرق التلوين التي قدمناها هنا تنطبق على أجهزة الكمبيوتر طراز IBM والأجهزة المتوافقة معه .

ومع ذلك فإذا كنت تملك جهاز كمبيوتر آخر فالاختلاف ليس كبيراً بين عبارات التلوين . فمع الأجهزة المنزلية غالباً لا تستخدم أمر مجموعات الألوان SETPAL وإنما تستخدم الأمر SETPC لتلوين القلم مباشرة . وكذلك الأمر SETBG لتلوين الخلفية .

والألوان المستخدمة تعتمد على نوع الجهاز فمثلاً مع الكمبيوتر المنزلي سنكثير توجد الألوان الآتية (على لوحة الأزرار) :

1	أزرق
2	أحمر
3	أرجواني (magenta)
4	أخضر
5	أزرق فاتح (cyan)
6	أصفر
7	أبيض
0	أسود

● مثال على الكمبيوتر سنكلير المنزلى :

SETBG 7 ← خلفية بيضاء
 SETPC 4 ← أخضر
 FD 50
 RT 90
 SETPC 2 ← أحمر
 FD 50
 RT 90
 SETPC 3 ← أرجوانى
 FD 50
 RT 90
 SETPC 1 ← أزرق
 FD 50
 HT ← إخفاء السلاحف

ونلاحظ هنا أن اللون البنى غير موجود بالكمبيوتر سنكلير لذلك استبدلناه باللون الأرجوانى (magenta) رقم 3 .

كما نلاحظ إضافة أمر إخفاء السلاحف كأمر أخير .

● فلنتقدم خطوة أخرى إلى الأمام :

لا يمانع « لوجو » أن يستقبل مجموعة من الأوامر فى سطر واحد . وفى هذه الحالة لن يتطلب رسم المربع الملون كل هذه السطور المتتالية ، كما أنك لن تضغط على الزر (Enter) إلا مرة واحدة . وعلى سبيل المثال يمكنك رسم مربع بمجموعة أوامر فى سطر واحد فقط أو فى سطرين كما يتراءى لك :

FD 50 RT 90 FD 50 RT 90
 FD 50 RT 90 FD 50

REPEAT

● ارسم مربعاً بأمر واحد فقط

لو ألقينا نظرة فاحصة على مجموعة الأوامر المتتالية التي ترسم المربع لوجدنا أنها عبارة عن تكرار لأمرين متتابعين هما (FD 50) ، (RT 90) .

وتمدنا لغة لوجو بإمكانية جديدة لتكرار الأوامر وهى الأمر REPEAT بمعنى كرّر وهو يستخدم كالآتي :

? REPEAT 4 [FD 50 RT 90]

عدد مرات التكرار

هذا الأمر يؤدي إلى تكرار مجموعة الأوامر التي بين القوسين المربعين ٤ مرات . جرب هذا الأمر سوف تشاهد المربع يرسم على الشاشة ثم تعود السلاحفاه إلى وضعها الأصلي .

والأمر REPEAT قد يحتوى بداخل القوسين المربعين على أية أوامر من أوامر لوجو بما في ذلك الأمر REPEAT نفسه .

FENCE, WRAP

● بناء سور حول السلاحفاه

رأينا أن السلاحفاه يمكن أن تتقبل أمراً للتحرك إلى الأمام مثل :

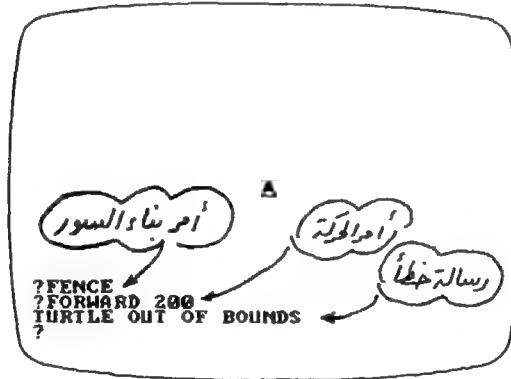
FD 1000

والرقم 1000 بالطبع كبير جداً ويخرج عن حدود أى شاشة . ولكن السلاحفاه إزاء هذا الأمر تقوم بعملية التحزيم التي ذكرناها وتعود من الناحية الأخرى للشاشة .

ويمكن إيقاف عملية التحزيم هذه بإعطاء الأمر FENCE بمعنى سور (الحديقة) ، أو السياج الذي يحدد مساحة من الأرض .

وهذا الأمر يجعل السلاحفاه تنقيد بحدود الشاشة في حركتها ولا تخرج عنها . وفي هذه الحالة . لو أنك أدخلت مع الأمر FD رقماً كبيراً يتعدى حدود

الشاشة فإن السلحفاه لا تتحرك ويرسل لك لوجو رسالة خطأ كالآتي :



ويمكن إلغاء هذه الخاصية بإعطاء الأمر WRAP فيعود الحال إلى ما كان عليه ولتجرب هذه الأوامر المتتابعة وشاهد النتائج .

```
FENCE
FORWARD 200
WRAP
FORWARD 200
```

● شاهد السلحفاه من النافذة WINDOW

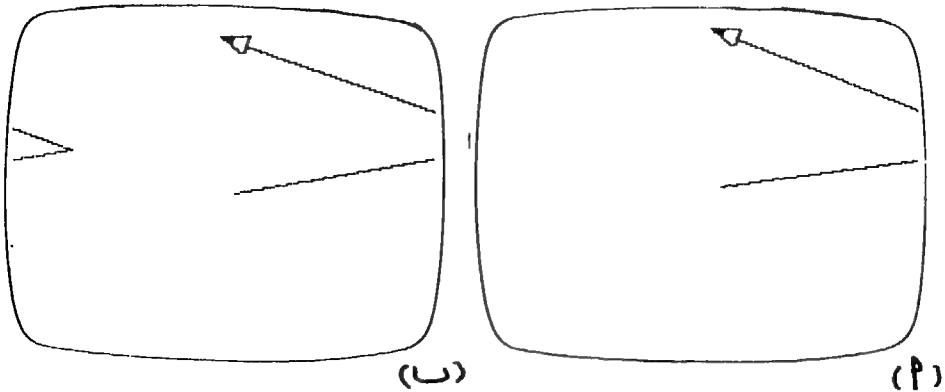
إذا أعطيت الأمر WINDOW فإن السلحفاه يمكنها الحركة خارج حدود الشاشة ، ولكنها متى تركت الشاشة فإنها تختفي . وبذلك تصبح الشاشة كأنها نافذة تحدّد المشهد أمامك .

جرب هذه المجموعة من الأوامر (يمكنك كتابتها مختصرة) :

```
CLEARSCREEN
WINDOW
RIGHT 80
FORWARD 180
LEFT 150
FORWARD 180
```

ونتيجة هذه الأوامر موضحة في الشكل على شاشة الكمبيوتر IBM. ثم جرب هذه المجموعة من الأوامر وفيها نستبدل الأمر WINDOW بالأمر WRAP الذي يعيد الحال إلى ما كان عليه فتبدأ خاصية التحزيم مرة أخرى. وتظهر النتيجة في الشكل (ب).

```
CLEARSCREEN
WRAP
RIGHT 80
FORWARD 180
LEFT 150
FORWARD 180
```



الشكل (أ) باستخدام الأمر WINDOW أما (ب) فباستخدام الأمر WRAP.

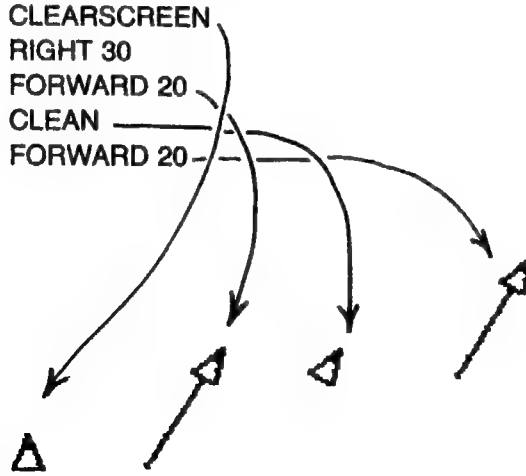
● تنظيف الشاشة بالأمر CLEAN :

هناك فارق أساسي بين تنظيف الشاشة بالأمر (CS) أو (CLEARSCREEN) وبين تنظيفها بالأمر (CLEAN) فهذا الأمر الأخير يسمح الشاشة من الرسم ولكنه يترك السلحفاه في وضعها الذي كانت عليه.

أما الأمر (CS) فهو يعيد كل شيء إلى أصله بما في ذلك وضع السلحفاه.

● مثال :

إدخل هذه المجموعة من الأوامر تشاهد النتائج المرسومة بجوارها بالتتابع :



HOME

● إعادة السلحفاه إلى بيتها

يمكنك بإعطاء الأمر HOME أن تعيد السلحفاه إلى بيتها الأصلي الذي تبدأ منه الحركة عادة ورأسها متجه إلى أعلى :

● مثال :

أدخل هذه المجموعة من الأوامر وشاهد النتائج :

CLEARSCREEN
FORWARD 50
HOME
CLEAN

هل لاحظت تأثير الأمر CLEAN الذي جاء بعد الأمر HOME ؟ إن هذه التركيبة من الأمرين CLEAN , HOME تكافئ تماماً الأمر CLEARSCREEN .

ولكنك لو أدخلت الأمر CLEAN قبل الأمر HOME فإن النتيجة سوف تختلف جرب وشاهد .

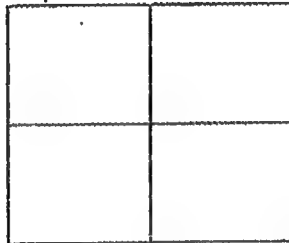
CLEARSCREEN
FORWARD 50
CLEAN
HOME

بالطبع هناك أوامر كثيرة مازالت في لغة « لوجو » للتحكم في السلحفاة ولكن بهذا القدر من الأوامر يمكنك أن ترسم ما شئت على الكثير من أجهزة الكمبيوتر .

تجربة



رسمنا من قبل الشكل المربع بأمر واحد باستخدام خاصية التكرار REPEAT . هل تستطيع أن ترسم الشكل الآتي المكوّن من المربعات باستخدام توليفة معينة من الأمر REPEAT مع أمر رسم المربع ؟ (حاول والإجابة في نهاية الكتاب) .



STARTROBOT, STOPROBOT

● التحكم في الروبوت

عندما يكون جهاز الكمبيوتر متصلاً بأحد الروبوتات الميكانيكية فإنه يمكن التحكم فيها بأوامر الحركة الخاصة بالسلحفاة هي :

FD , BK , PU , PD , RT , LT

ويتم ذلك بإعطاء الأمر :

STARTROBOT

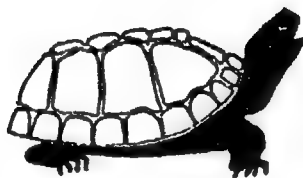
وعند إعطاء هذا الأمر يجب أن يكون قد سبق إعداد برنامج الحركة الذي يتحكم في الروبوت . فإذا لم يكن البرنامج موجوداً فإن الكمبيوتر يبحث عنه ويحاول تحميله من القرص المغنطيسي أو الشريط الكاسيت .

وهذا الأمر قد يختلف من جهاز إلى آخر ولكن عند الحاجة إليه يسهل البحث عن أمر مماثل في كتالوج استخدام الجهاز بلغة لوجو .

ولإيقاف التحكم في الروبوت يستخدم الأمر :

STOPROBOT

فيؤدي إلى إنهاء العلاقة مع الروبوت الميكانيكي واقتصار الأوامر على حركة السلحفاة .



٣

تعليم الكمبيوتر
أوامر جديدة

TO, END

● علم الكمبيوتر أمراً جديداً

رغم الاختصارات التي استخدمناها مع أوامر لوجو ، ورغم الوسائل الموفرة للوقت مثل تكرار الأمر باستخدام أمر التكرار REPEAT ولكن مع هذا كله فإن إدخال عدة أوامر متتابعة للكمبيوتر يصبح عملاً مملاً بعد فترة من إيجادته .

فلو أردنا أن نرسم صندوقاً مربعاً فإننا نضطر في كل مرة أن نكتب مجموعة من الأوامر مثل :

? FD 50 RT 90

? FD 50 RT 90

? FD 50 RT 90

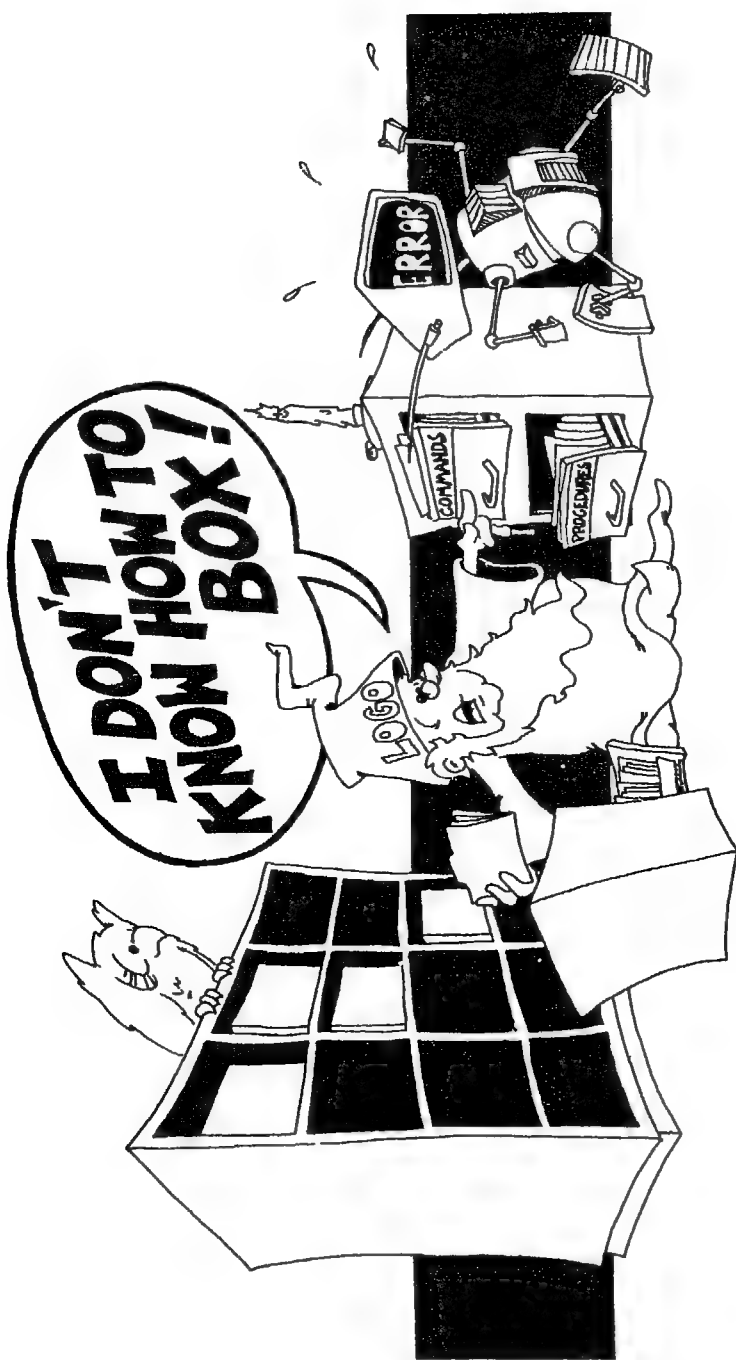
? FD 50 RT 90

ألا توجد طريقة أخرى لكي نعطي للكمبيوتر أمراً واحداً مختصراً مثل

BOX فيرسم الصندوق المطلوب ؟

ولكننا نعلم أننا لو أعطينا الكمبيوتر أمراً لا يعرفه مثل BOX فسوف يشكو الساحر لوجو قائلاً : « أنا لا أعرف كيف أرسم صندوقاً » وبلغته الإنجليزية الطريقة يقول :

“I don't know how to box” !



الساحر لوجو .. يفتش كل خزائنه بحثاً عن أمر اسمه BOX
ولكن بلا جدوى ..

ومع ذلك يمكننا تنفيذ ما نريد بطريقة ما .

فهناك وسيلة لحفظ مجموعة الأوامر التي ترسم الصندوق في ذاكرة الكمبيوتر فيما يسمى بالبرنامج الفرعي^(١) أو البرنامج (Procedure) .

ويعم ذلك بإعطاء الأمر الآتي (على شاشة الكتابة TEXTSCREEN) :

? TO BOX

>

اسم البرنامج

عند إعطاء هذا الأمر تظهر العلامة (>) عند أول السطر ويتوقف الكمبيوتر عن تنفيذ أوامر لوجو ويتوقف أيضاً عن الاعتراض إذا أنت أخطأت في الكتابة بلغة لوجو . فهو يمنحك الآن صفحة تكتب فيها ما تشاء . وبعد أن تنتهي من الكتابة سوف يحفظ كل ما كتبه في الذاكرة تحت الاسم

BOX وبذلك فإن البرنامج BOX بعد إتمامه . سوف ينضم إلى مجموعة الأوامر التي يفهمها وينفذها الكمبيوتر مثل RT , BK , FD ... إلخ . ولذلك لا يجوز أن تطلق على برنامجك اسماً من هذه الأسماء التي يفهمها الكمبيوتر . فهذه الأسماء في الحقيقة هي برامج أيضاً ولكن سبق تخزينها في لغة لوجو ، وهي تسمى البرامج الأولية (primitive procedures) . فلنبدأ الآن بكتابة البرنامج BOX ولا تنس أن تضغط على الزر Enter بعد كل سطر .

TO BOX

> FORWARD 50

> RIGHT 90

> FORWARD 50

> RIGHT 90

> FORWARD 50

> RIGHT 90

> FORWARD 50

> RIGHT 90

> END

اسم البرنامج

نهاية البرنامج

(١) يستخدم أيضاً مصطلح الروتين (routine)

وبعد أن تفرغ من كتابة سلسلة الأوامر التي ترسم الصندوق يجب أن تتبعها بالأمر END . هنا فقط يستيقظ اللوجو من جديد ويعرف أنك انتهيت من الكتابة وأنه جاء دوره ليبدأ في تنفيذ الأوامر لذلك تظهر علامة الاستفهام مرة أخرى على يسار الشاشة بعد أن يرسل إليك الرسالة الآتية .

BOX DEFINED

بهذه الرسالة الأخيرة يصبح البرنامج BOX أمراً من أوامر لوجو المحفوظة في ذاكرته ويمكنك تنفيذ جميع خطوات البرنامج بالأمر التالي :

? BOX

بمجرد أن تضغط على الزر Enter تبدأ السلحفاة في رسم الصندوق المربع بشرط أن يكون البرنامج سليماً وخالياً من الأخطاء . فإذا لم يكن كذلك فإن الكمبيوتر يتوقف عند الخطأ ويرسل لك شكوى بما أعاقه عن العمل .

فما هو التصرف في حالة وجود خطأ بالبرنامج ؟

الأمر البديهي هو إعطاء الأمر BOX TO مرة أخرى وكتابة البرنامج من جديد . ولكن لا .. هناك حل أفضل بكثير .

EDIT

● إصلاح برنامج لوجو

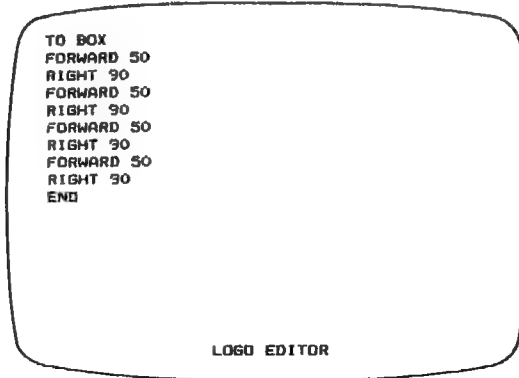
يمكنك استرجاع البرنامج BOX باستخدام وسيلة داخلية ضمن لغة لوجو تسمى « المحرر » (Screen-Editor) .

ويستدعى المحرر محتويًا على البرنامج BOX بالأمر :

? EDIT "BOX"

بمجرد كتابة هذا الأمر تجد البرنامج BOX معروضاً على الشاشة مسبقاً بكلمة BOX TO وجاهزاً على الإصلاح .

والمحرر لا يراجع الأخطاء ولا ينفذ الأوامر أيضاً ، فهو مجرد وسيلة للكتابة أو لإصلاح الكتابة السابقة . ويمكنك بداخله أن تتحرك في جميع الاتجاهات باستخدام الأسهم . كما يمكنك استخدام أزرار المسح (Del.) وحشر الحروف (ins.) وسائر أزرار الكتابة المعروفة .



محرر لوجو يعرض البرنامج BOX

ويقال أن الكمبيوتر في هذه الحالة موجود في طور التحرير (edit made) أما الطور العادى الذى عرفناه من قبل فهو طور الأوامر (Command mode) ويتميز بعلامة الاستفهام التى تظهر على اليسار . وللخروج من طور التحرير إلى طور الأوامر يلزم استخدام زر خاص . ففي الكمبيوتر IBM والطرازات المشابهة نستخدم الزر Esc . وفي الأجهزة الأخرى توجد عادة أزرار مشابهة فى الوظيفة وربما تحمل أسماء أخرى . ففي الكمبيوتر سنكلير مثل يلزم الضغط على عدة أزرار هى :

symbol shift + caps shift + C

وبصفة عامة يتم الرجوع لتعليمات استخدام برنامج لوجو مع الأجهزة الصغيرة . وعند الخروج من المحرر سوف تشاهد الرسالة :

BOX DEFINED

← ?

علامة الاستعداد فى طور الأوامر

بهذا يكون البرنامج محفوظاً في ذاكرة الكمبيوتر كأمر من أوامره المعروفة مسبقاً ويمكنك في هذه الحالة رسم الصندوق باستخدام الأمر البسيط BOX .

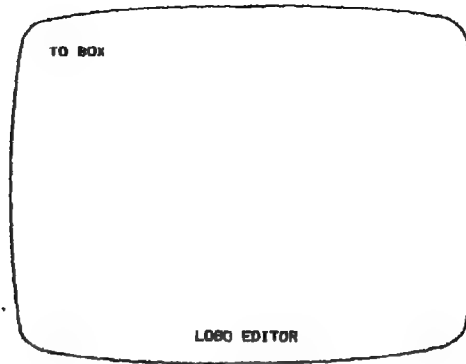
● كتابة البرنامج باستخدام المحرر

فضلاً عن إصلاح البرنامج باستخدام المحرر فإنه يمكن من الأصل كتابة البرنامج الجديد باستخدام المحرر . ويتم ذلك باستدعاء المحرر بالأمر :

EDIT "BOX



عندئذ سوف تشاهد العبارة BOX TO مكتوبة أعلى الشاشة أما بقية الشاشة فهي جاهزة على الكتابة .



شاشة محرر لوجو جاهزة على الكتابة

بعد ذلك تبدأ في كتابة البرنامج سطراً بسطر مع الضغط على الزر Enter بعد كل سطر (وهذا موضح بالعلامة (␣) المرسومة على زر الكمبيوتر IBM) :

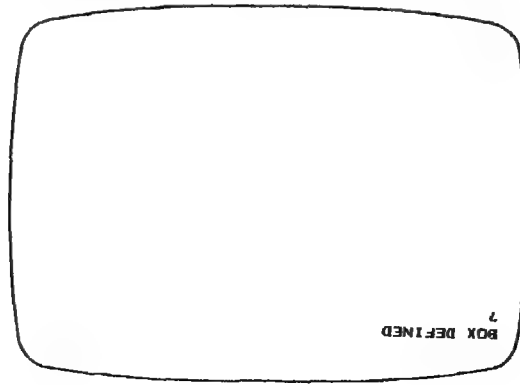

```

FORWARD 50 ←
RIGHT 90 ←
FORWARD 50 ←
RIGHT 90 ←
FORWARD 50 ←
RIGHT 90 ←
FORWARD 50 ←
RIGHT 90 ←
END ←

```

اضغط
Enter

بعد كتابة البرنامج يتم الخروج من طور التحرير على الزر ESC (أو ما يعادله في الأجهزة الأخرى) . فتظهر الرسالة BOX DEFINED أعلى الشاشة كالآتي :



تم تعريف البرنامج BOX

بهذا يكون البرنامج BOX جاهزاً على التنفيذ

وبالتبع في حالة حدوث خطأ بالبرنامج يمكن استدعاء البرنامج BOX بنفس الطريقة وإصلاحه .

كما يمكن مسح البرنامج BOX من ذاكرة الكمبيوتر بالأمر :

ERASE "BOX

ER "BOX

أو بالصورة المختصرة :

ويمكن تخزين أكثر من برنامج في نفس الوقت في ذاكرة الكمبيوتر بنفس

الطريقة ، حيث تخصص لغة لوجو مساحة من الذاكرة تسمى حيز العمل (working space) وفي هذا الحيز تخزن البرامج التي يكتبها مستخدم اللغة وهذه خاصية فريدة للغة لوجو .

ويمكن مسح جميع البرامج من حيز العمل بالأمر :

ERALL

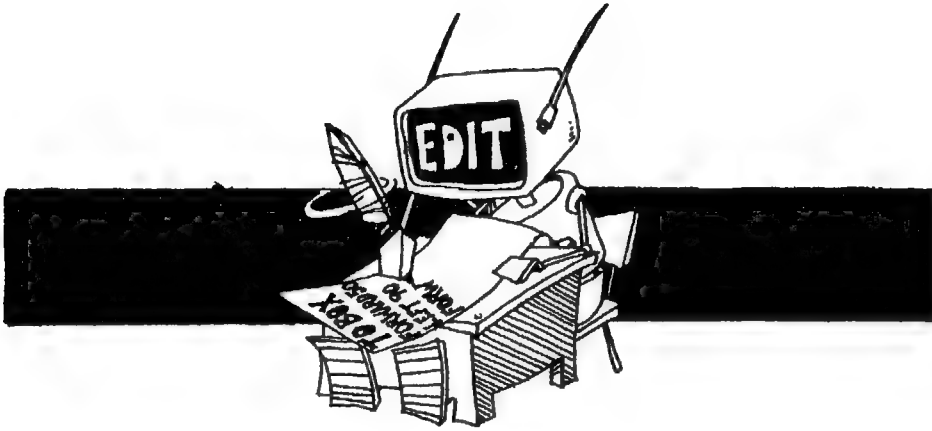
● ماذا يحدث بداخل الكمبيوتر ؟

فيما يلي سوف يساعدنا الساحر لوجو على توضيح بعض التفاصيل التي تحدث بداخل الكمبيوتر عند استخدام المحرّر Editor لكتابة وحفظ البرنامج BOX ثم تنفيذه في النهاية .

ويعاون الساحر لوجو كل من الروبوت EDIT الذى يمثل المحرّر والروبوت BOX الذى يمثل البرنامج .



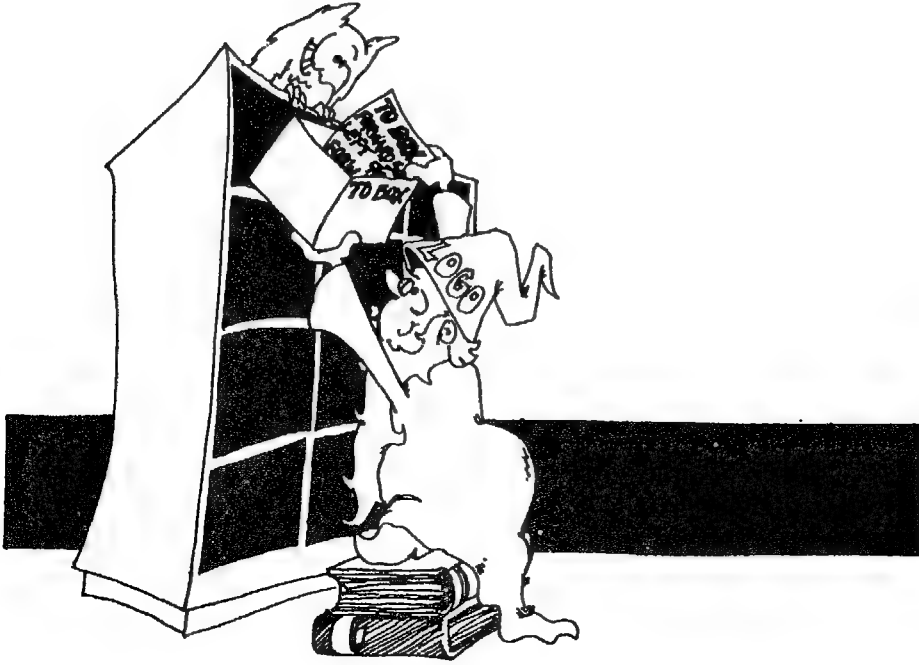
(أ) الساحر لوجو يستدعى الروبوت EDIT ويخبره باسم البرنامج المطلوب إنشاءه BOX .



(ب) الروبوت EDIT يختزن كل ما نكتبه من أوامر وكل ما نضيفه من تعديلات بصفحة المحرر .



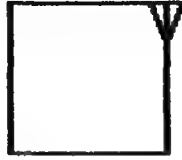
(ج) عندما نخرج من المحرر إلى طور الاوامر فإن الروبوت EDIT يسلم البرنامج BOX إلى الساحر لوجو ليحتفظ به .



(د) يقوم الساحر لوجو بتخزين تعليمات البرنامج BOX في مكان معروف
يسهل استرجاعها منه . لذلك نجدها يضعها في علبة مكتوب عليها "TO
BOX".



(٥) عندما نعطى الأمر BOX يقوم الساحر لوجو باسترجاع التعليمات المخزنة في العلبة : "TO BOX"



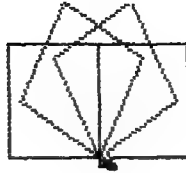
نتيجة تنفيذ الأمر BOX

وهذا تلخيص بالأوامر التي تعلمناها حتى الآن مع الاختصارات الممكنة :

الوظيفة	الاختصار	الأمر
لبءء ءءلم الكومبوءر .	—	TO
إنءءاء ءءلم للكومبوءر .	—	END
اسءءءاء مءرر لوءوء .	ED	EDIT
لمسء برءاء أو أكءر .	ER	ERASE
لمسء كل البراءء .	—	ERALL

● اكءب لغة الكومبوءر بءفسك :

بعء نءاءنا فى ءلقىن الكومبوءر الأمر الءءىء BOX يمءننا أن نءكءب لغة ءءىءة ءمءوى على أوامر لىسء موءوءة فى لغة لوءوء أصلاً ولكنها موءوءة فى ذاكرة الكومبوءر الءى نعمل علىه .



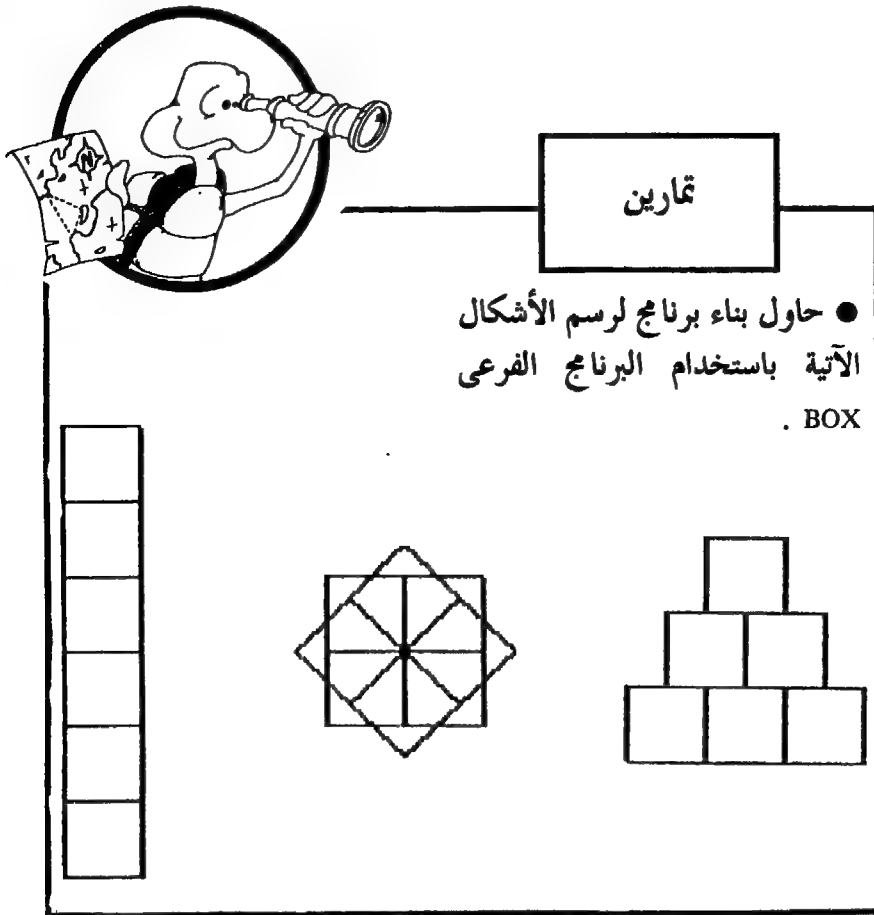
شكل مرسوم باسءءءام الأمر BOX

فالرسم الموءوء بالشكل ءم ءنفىءه بالبرءاء الآءى :

PEPEAT 4 [BOX LT 30]

وهنا قد ظهرت كلمة BOX ضمن برنامج آخر متداخلة مع أوامره وكأنها جزء من لغة لوجو . فما معنى ذلك ؟

في الحقيقة أنه طالما كان البرنامج BOX مخزناً في ذاكرة الكمبيوتر فيمكنك استخدامه كما شئت سواء كأمر مباشر أو كأمر داخل برنامج آخر كهذه الحالة . وعندما يستخدم البرنامج BCX بهذه الصورة فإنه يسمى برنامجاً فرعياً (subprocedure) ويسمى البرنامج الذي يحتويه البرنامج الذي يحتويه بالبرنامج الرئيسي (superprocedure) .



PO, POALL, POTS

● عرض البرامج على الشاشة

● إذا أعطيت الأمر :

PO "BOX

فإن الكمبيوتر يعرض أمامك كل خطوات البرنامج BOX على الشاشة بحيث يمكنك مراجعتها مراجعة سريعة بدون الدخول في طور التحرير . والأمر PO هو اختصار للعبارة (PRINT OUT) .

● وإذا كان في الذاكرة أكثر من برنامج فيمكنك عرضهم جميعاً بأمر واحد هو :

POALL

● كما يمكنك عرض عناوين البرامج فقط (أسمائها) بالأمر :

POTS

والكلمة POTS هي اختصار العبارة (PRINT OUT TITLES) .

ED, ED [....]

● إمكانات أخرى للمحرر

● علمنا أنه يمكن استدعاء برنامج ما على شاشة المحرر بكتابة الأمر EDIT أو الصورة المختصرة ED يليها اسم البرنامج مثل :

ED "BOX

● يمكن أيضاً إعطاء الأمر EDIT بدون أى اسم معه وفي هذه الحالة يستدعى المحرر آخر برنامج سبق استدعاؤه

- من الممكن أيضاً استدعاء أكثر من برنامج وإصلاحهم معاً بوضع هذه الأسماء في قائمة كالآتي :

ED [BOX STAR PYRAMD]

والأسماء التي بداخل القوسين المربعين هي أسماء البرامج المطلوب إحضارها على شاشة المحرر لإصلاحها .

- حفظ البرامج في الذاكرة الخارجية واستدعاؤها :

SAVE, LOAD, DIR, ERASEFILE

كما نعلم أن ذاكرة الكمبيوتر التي نتحدث عنها دائماً هي ذاكرة مؤقتة أو متطايرة بمعنى أن كل ما هو موجود بها يتطاير منها بمجرد إطفاء الجهاز . فإذا أطفأنا الكمبيوتر ذهبت جميع البرامج وذهبت لغة لوجو أيضاً . والاسم الدقيق للذاكرة هو الذاكرة الداخلية .

ولحفظ البرامج بصفة مستديمة يلزم حفظها في الذاكرة الخارجية التي تتميز بالدوام والثبات وهي القرص المغنطيسي أو الشريط الكاسيت .

ولحفظ برنامج ما يدعى SALLY نعطي الأمر :

SAVE "SALLY"

ولاستدعاء هذا البرنامج من القرص أو الكاسيت نعطي الأمر :

LOAD "SALLY"

وهذه الأوامر تنطبق على القرص المغنطيسي للكمبيوتر IBM وكذلك على الشريط الكاسيت للأجهزة الصغيرة . أما أوامر القرص المغنطيسي للأجهزة الصغيرة فقد تختلف ويرجع إليها بدفتر اللغة .

يمكن أيضاً عرض جميع البرامج المخزنة على القرص باستخدام الأمر :

DIR

وفي بعض الأجهزة الأخرى يتم عرض البرامج باستخدام الأمر :

CATALOG

كما يمكن مسح البرنامج من القرص المغنطيسي باستخدام الأمر .

ERASEFILE "SALLY

كما يمكن حفظ واستدعاء الصور المرسومة بلغة لوجو مباشرة بالأوامر :

SAVEPIC "BOXES

للحفظ

للاستدعاء

LOADPIC "BOXES

اسم الصورة

وبصفة عامة فإن أوامر الحفظ ليست قياسية ويلزم الرجوع لكتاب الجهاز المعين .

SAVE "LPT1, DRIBBLE,

● تشغيل جهاز الطباعة

NODRIBBLE

PRINTON, PRINTOFF, OUTDEV

هذه هي أوامر الطباعة على جهاز الطباعة PTINYRT للكمبيوتر IBM :

SAVE "LPT1

هذا الأمر يطبع كل شيء في حيز العمل على الطباعة. أما الأمر التالي فهو يجعل الطباعة على اتصال دائم بالكمبيوتر :

DRIBBLE "LPT1

لذلك يقال أن الطباعة في حالة « تصنّت » ! فإذا كتبت الأمر الآتي :

PRINT [hello you]

فإن العبارة hello you تطبع فوراً على الطابعة .
وإذا أعطيت الأمر :

POTS

فإن أسماء البرامج التى فى الذاكرة الداخلية تُطبع فوراً .
وإذا كتبت الأمر :

PO "BOX

فإن سطور البرنامج BOX تُطبع على الورق .
وهكذا ..

والآن نعطي الأمر المضاد الذى يوقف عمل الطابعة وهو :

NODRIBBLE

ملاحظة

أما أوامر الطبع للأجهزة الصغيرة فقد تختلف كثيراً وعلى سبيل المثال
فالكومبيوتر سنكلير يستخدم الأمر PRINTON كنظير للأمر DRIBBLE
ويستخدم PRINTOFF كنظير للأمر NODRIBBLE أما لنسخ الشاشة
كلها فهو يستخدم الأمر COPYSCREEN وهذا يناظره فى الكومبيوتر
IBM الضغط على الأزرار :

Shift ↑ + Prtscr.

أما مع الكومبيوتر apple II فالأمر 1 OUTDEV يستخدم لتشغيل
الطابعة والأمر 0 OUTDEV يستخدم لفصل الطابعة . وبصفة عامة
يرجع لكتاب اللغة .

٤

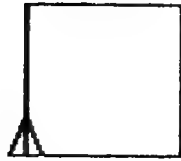
تطبيقات مختلفة
للرسم بالسلحفاة

● استخدام البرامج الفرعية :

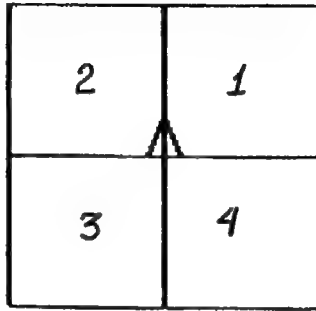
إذا كنت قد احتفظت بنسخه من برنامج رسم المربع يمكنك الآن أن تستدعيها من الشريط الكاسيت أو من القرص المغنطيسي فإذا لم تكن قد فعلت فاكتب هذا البرنامج ؛ وتأكد من أنه يرسم الشكل المطلوب الموضح بعد :

```
TO SQUARE  
FORWARD 50  
RIGHT 90  
FORWARD 50  
RIGHT 90  
FORWARD 50  
RIGHT 90  
FORWARD 50  
RIGHT 90  
END
```

البرنامج الفرعي



نريد الآن استخدام هذا البرنامج كبرنامج فرعي لرسم أربعة مربعات متلاصقة ببعضها البعض كما في الشكل التالي :



هذا يمكن أن يتم بإنشاء برنامج جديد (رئيسي) يستخدم البرنامج الفرعي SQUARE بالطريقة المناسبة . وهي رسم المربع رقم 1 ثم إدارة السلحفاة ٩٠:

درجة جهة اليسار (أو جهة اليمين) ورسم المربع رقم 2 ثم إدارتها ٩٠ درجة أخرى لليساار ورسم المربع رقم 3 وهكذا بذلك يحتوى البرنامج الرئيسى على الأوامر التالية :

TO BOXES
SQUARE
LEFT 90
SQUARE
LEFT 90
SQUARE
LEFT 90
SQUARE
LEFT 90
END



وبالطبع سوف يقول أصدقاؤنا الذين استوعبوا الأجزاء السابقة :
« ولكن هذه طريقة بدائية لكتابة البرامج .. أين REPEAT وأين الاختصارات لتغنى عن هذا كله ؟ » وهذا حق .
فمن الممكن كتابة البرنامج الأول كالاتى :

REPEAT 4 [FD 50 RT 90]

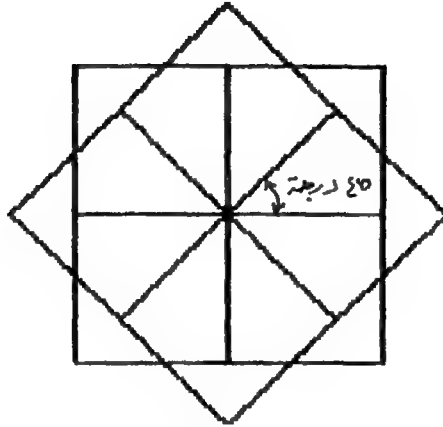
والبرنامج الرئيسى يصبح كالاتى :

REPEAT 4 [SQUARE LT 90]

ولكن لا بأس من استخدام البرامج البدائية أحياناً حتى يتابعنا جميع القراء وسوف نقدم الصورة المختصرة من أن إلى آخر .

والآن بإعطاء الأمر BOXES فإن البرنامج الرئيسى BOXES يستدعى البرنامج الفرعى SQUARE ليرسم المربعات الأربعة . ثم تعود السلحفاه إلى بيتها وفي اتجاهها الأصلي .

نريد الآن أن نستخدم البرنامج الرئيسى BOXES كبرنامج فرعى لرسم الشكل الموضح بعد والمكون من تكرار للشكل السابق بزوايا مختلفة :

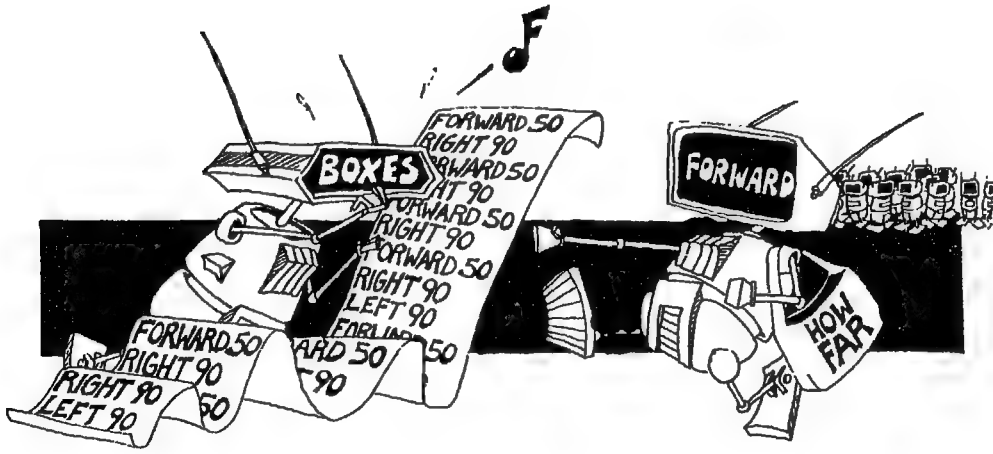


هذا الشكل عبارة عن الشكل الأصلي BOXES مرسوماً مرتين . ولكن في المرة الثانية رسمناه مائلاً بزاوية ٤٥ درجة .

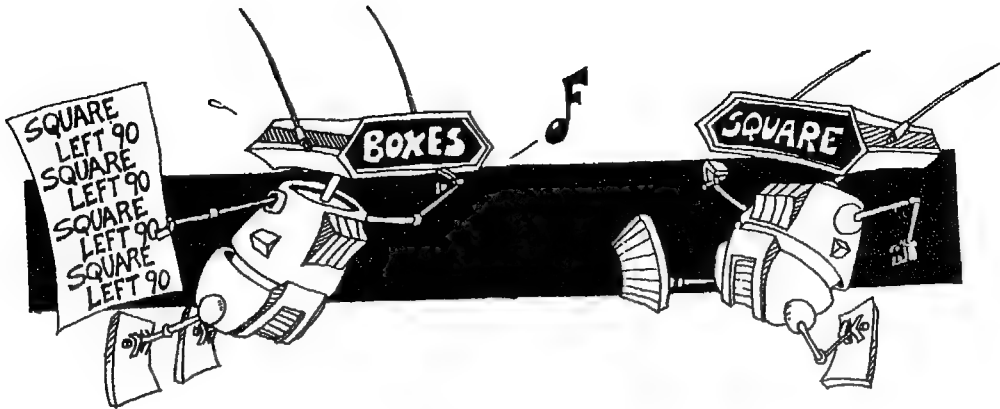
لنكتب إذن هذا البرنامج الرئيسى وليكن اسمه STAR ليستدعى البرنامج الفرعى BOXES ويستخدمه في رسم الشكل .

```
TO STAR
BOXES
RIGHT 45
BOXES
END
```

تصور الآن لو لم نستخدم البرامج الفرعية ! كم كان البرنامج الأخير سيكون طويلاً ومرهقاً في كتابته بل مرهقاً في قراءته وفهمه أيضاً . وهذا هو الربوط BOXES حائر ما بين الأوامر الكثيرة التى يحتوى عليها برنامج بلا برامج فرعية .



كم يكون العمل مرهقاً بدون البرامج الفرعية !



وكم يصبح العمل سهلاً باستخدام البرامج الفرعية

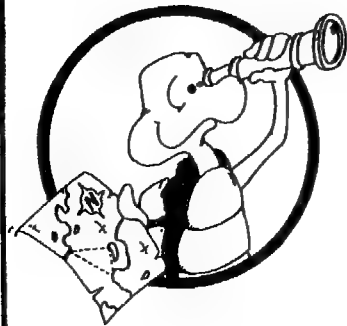
وهذه هي الصورة المختصرة للبرامج الثلاثة (مع إضافة الرقم 1 للأسماء للتمييز):

```
TO SQUARE 1  
REPEAT 4 [FORWARD 50 RIGHT 90]  
END
```

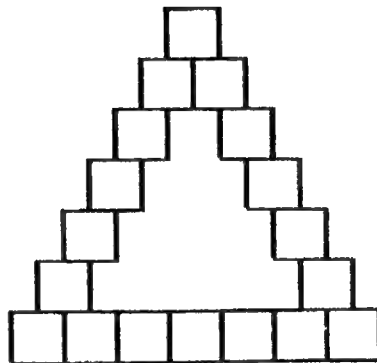
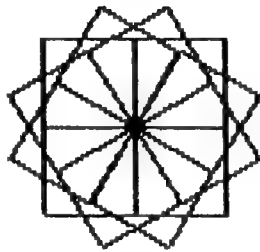
```
TO BOXES1  
REPEAT 4 [SQUARE1 LEFT 90]  
END
```

```
TO STAR1  
REPEAT 2 [BOXES1 RIGHT 45]  
END
```

تجارب



والآن حاول رسم هذه الأشكال
مستخدماً البرنامج الفرعي SQUARE
أو البرنامج الفرعي BOXES .



● رسم المثلثات :

عندما رسمنا شكلاً مربعاً استخدمنا الدوران بزاوية ٩٠ درجة وبذلك فإن الزاوية الدائرية الكاملة (٣٦٠ درجة) اتسعت لعدد ٤ مربعات لأن $360 = 90 \times 4$.

عندما نرسم مثلثاً متساوي الأضلاع فإن كل زاوية من زواياه تكون ٦٠ درجة . أى أن السلحفاه يجب أن تستدير ١٢٠ درجة بعد انتهائها من رسم كل خط .



إذن يمكن كتابة برنامج المثلث كالاتى :

```
TO TRIANGLE
REPEAT 3 [FORWARD 80 RIGHT 120]
END
```

التكرار

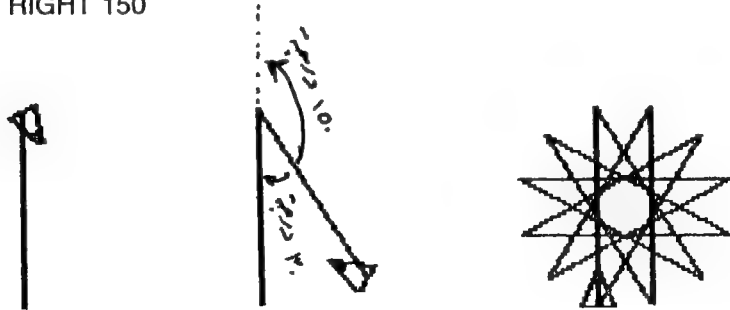
زاوية الدوران

ومن هذا الشكل يمكن أن ننشئ العديد من الأشكال باستخدامه كبرنامج فرعى . وسوف نتعرض لكثير من تطبيقات المثلثات فى الأجزاء القادمة . ولكننا الآن نريد أن نتقل إلى نوعية أخرى من المثلثات وهى النجوم .

● رسم النجوم :

لنبدأ محاولة رسم نجمة بالسطور الآتية وباستخدام الدوران ١٥٠ درجة .

FORWARD 50
RIGHT 150
FORWARD 50
RIGHT 150



خطوات رسم النجمة

وبتكرار هذا الرسم مرة بعد مرة نحصل على الشكل المطلوب .

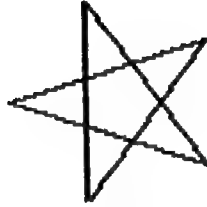
والواقع أنه يمكن تجربة عدة زوايا مختلفة بخلاف ١٥٠ درجة وملاحظة ما يحدث على الشاشة ثم تقليل أو تكبير الزاوية بحيث تلتقى نقطة البداية مع نقطة النهاية . أما عدد مرات التكرار فهي تعبر عن عدد الخطوط المكونة منها النجمة وهي ١٢ خطأً (في حالتنا هذه) أى أن البرنامج يمكن أن يكون كالآتي :

TO STAR 150
REPEAT 12 [FORWARD 100 RIGHT 150]
END

(زاوية الدوران) (عدد الخطوط)

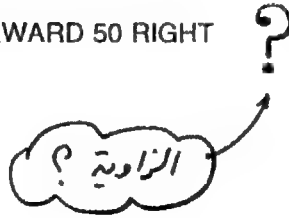
هذه نجمة ذات ١٢ ضلعاً .

فماذا لو أردنا رسم النجمة العربية ذات الخمسة أضلاع ؟

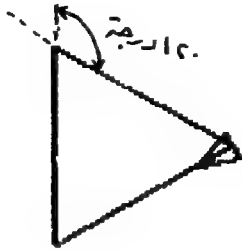


من المتوقع أن يكون البرنامج بالصورة التالية :

REPEAT 5 [FORWARD 50 RIGHT ?]



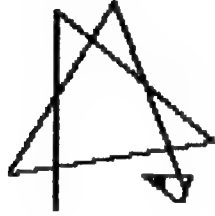
ومبدئياً فإن هذه الزاوية لابد أن تقع بين ٩٠ درجة و ١٨٠ درجة ولكن لإيجادها بدقة فهذا يحتاج إلى تجارب كثيرة . فلنبدأ التجربة بالزاوية ١٢٠ درجة .



REPEAT 5 [FORWARD 80 LEFT 120]

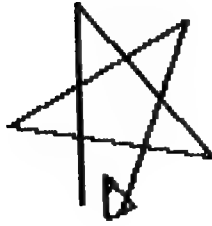
إن نقطتي البداية والنهاية تلتقيان ولكن الشكل الناتج مثلث كما نرى ! إن الزاوية ١٢٠ درجة تبدو صغيرة جداً لأنها لا تجعل الخطوط تتقاطع . فلنجرب إذن الزاوية ١٣٠ درجة .

REPEAT 5 [FORWARD 80 LEFT 130]



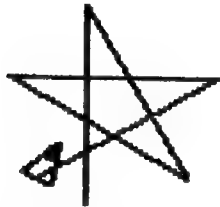
إن التقاطع يحدث ولكن الشكل مفتوح لا تلتقى بدايته مع نهايته .
فلنجعلها ١٤٠ درجة .

REPEAT 5 [FORWARD 80 LEFT 140]



مازال الشكل مفتوحاً لكنه يشير بقرب النجاح . لنجرب الآن ١٥٠
درجة .

REPEAT 5 [FORWARD 80 LEFT 150]



لقد حصلنا على الشكل المقفول .. ومع ذلك فنقطة التقاء البداية مع النهاية
حدثت « متأخرة » قليلاً . إذن هذه الزاوية أكبر من المطلوب . فالزاوية

المطلوبة تقع بين ١٤٠ درجة و ١٥٠ درجة . لكي تكمل التجارب لابد من إخفاء السلحفاه حتى تشاهد نقطة التلاقى بدقة . ربما يحتاج الأمر إلى تجارب كثيرة .. والبديل الوحيد أن نلجأ إلى التخمين الرياضى .. فى الفقرة القادمة ..

* هل هناك بديل للتجربة والخطأ ؟

القيمة التى نبحث عنها هى 144 .

ولكنها قيمة صعبة المثال مع المحاولة والخطأ .

وفى الواقع أن هناك طريقة تفكير رياضية تغنى عن هذه المحاولات العشوائية .

إن السلحفاه فى رحلتها لرسم أى شكل مقفول ، تدور خلال زاوية مقدارها أحد مضاعفات العدد 360 (بمعنى أن تكون 360 أو 2×360 أو 3×360 وهكذا ..) وذلك قبل أن تصل إلى نقطة البداية .

وفى حانة النجمة ذات الخمسة أضلاع فإن السلحفاه تدور خمس مرات . فليكن إذن مقدار كل زاوية دوران هو $(360/5)$ وهذه هى التجربة الأولى . فإذا لم تنجح لنجرب الرقم الآخر وهو $(360/5) \times 2$ وهكذا ... والرقم الذى سينجح هو الأخير أى 144 درجة .

وعلى ذلك يكون البرنامج هو :

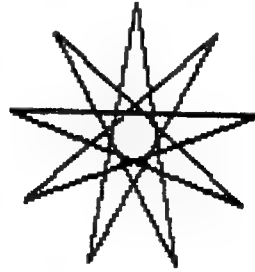
REPEAT 5 [FD 80 LT 144]

ولنسم هذا البرنامج الفرعى STAR144 دلالة على الزاوية المحيرة 144 .

تجربة



استتج الآن الزاوية التي تدور
بها السلحفاة لرسم النجمة ذات
الأضلاع التسعة الموضحة بالرسم
واكتب البرنامج الذي يرسمها .



mirror

* المرآة

لو أنك استبدلت كل دوران لليمين بدوران لليسار (والعكس) فإنك تحصل
على نفس الشكل المرسوم ولكنه يبدو كما لو كان معكوساً على سطح مرآة .

فالمربع square سبق لنا أن رسمناه بالبرنامج الآتي :

REPEAT 4 [FD 50 RT 90]

والمربع المنعكس على المرآة يكون برنامجه كالتالي :

REPEAT 4 [FD 50 LT 90]

ويطلق على المربع الأخير صورة المربع على المرآة (mirror image) والشكل
التالي يوضح بعض الرسومات التي رسمناها وصورة كل رسم على المرآة .
وعليك باستنتاج البرامج المناظرة .

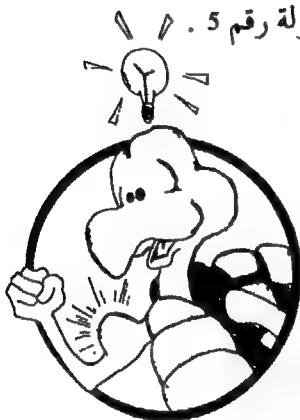


وهناك طريقة أخرى للحصول على الصورة وهي استبدال الزاوية المقررة للدوران ولتكن n بالزاوية $(360-n)$ مع بقاء جهة الدوران كما هي . فمثلاً للحصول على صورة المربع نستبدل الزاوية 90 بالزاوية 270 وللحصول على صورة المثلث نستبدل الزاوية 120 بالزاوية 240 وهكذا ..

● فكرة لهواة البحث والتقيب .. فقط !

البرنامج الذى عرضناه لرسم النجمة ذات الإثنى عشر ضلعاً استخدمنا فيه الزاوية ١٥٠ درجة . فإذا أردنا أن نعرف كيفية حساب هذه الزاوية . فهي حسب القاعدة تكون أحد مضاعفات الزاوية $(360/12)$ أى الزاوية (30) درجة . وبالتجربة نجد أنها $5 * (360/12)$ أى أنها المحاولة رقم 5 .

والسؤال هنا :



ماذا عن الأشكال الأخرى التى
تناظر المحاولة الأولى والثانية والثالثة
والرابعة التى تناظر الزاوية
 $4 * (360/12)$ ، $3 * (360/12)$..
وهكذا ..

ولنعتبر مثلاً الشكل الذى نحصل
عليه عند الزاوية $3(360/12)$ أى ٩٠
درجة .

إن هذا الشكل عبارة عن مربع وهذا بدىي لأن الدوران بزاوية ٩٠ درجة
مرة بعد مرة يرسم مربعاً .

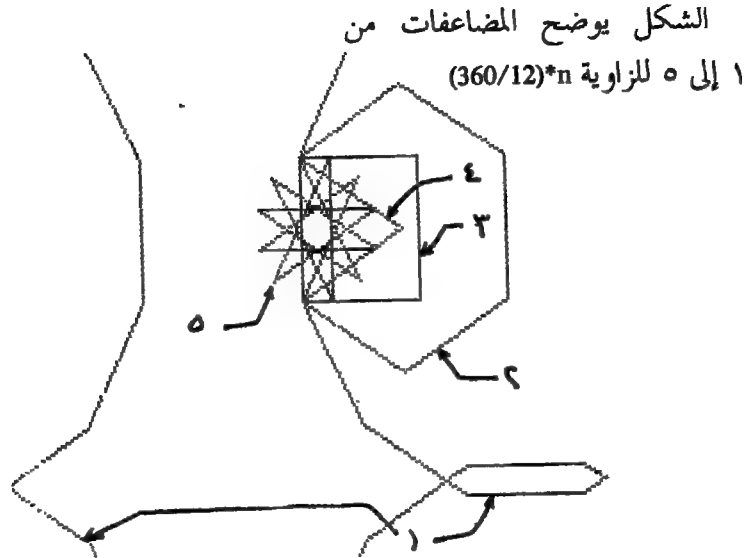
ولكن البرنامج يرسم هذا المربع ٣ مرات لأن أمر التكرار يعقبه الرقم ١٢ .

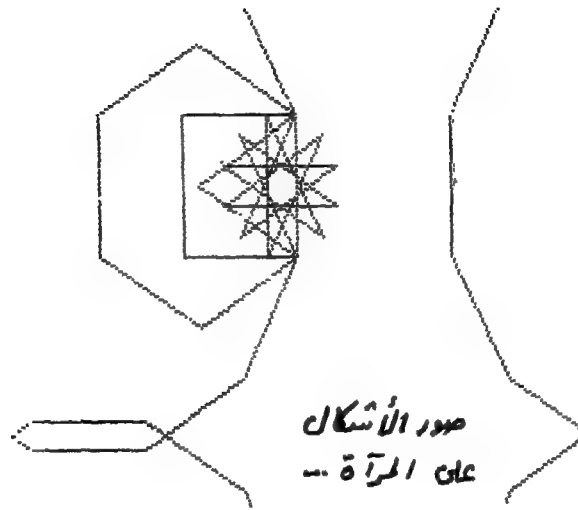
REPEAT 12 [FD 50 RT 90]

ولو أكملنا التجارب حتى المضاعفات رقم ٦ ، ٧ ، ٨ ، ٩ ، ١٠ ،
١١ .. لوجدنا أن الأشكال التى نحصل عليها تتكرر وأحياناً تنقلب إلى صور
على المرآة .

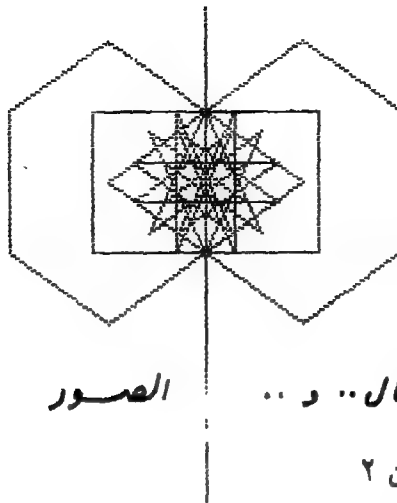
والآتى بعد هو بعض الأشكال التى نحصل عليها بتغيير الرقم المضروب فى
 $(360/12)$ وكذلك صورها على المرآة .

جرب البرنامج وحاول أن تعلق لهذه الأشكال والصور التى تحصل عليها فى
ضوء الدراسة السابقة .





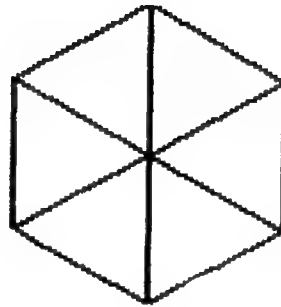
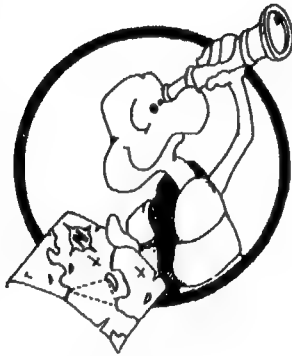
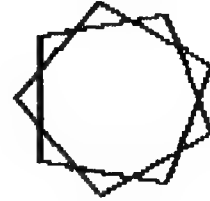
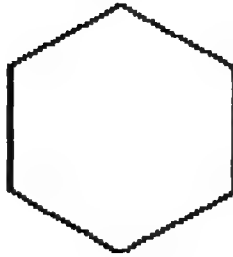
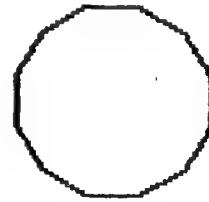
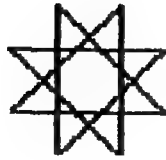
الشكل يوضح الصور المناظرة
للأشكال السابقة على المرآة وهي
تناظر المضاعفات من ٧ إلى ١١ .



الشكل يوضح المضاعفات من ٢
إلى ٥ وصورها على المرآة والخط
الفاصل بين المجموعتين يوضح المرآة
وهو نفسه أحد الأشكال الناتجة عن
القيمة (0 = n) .

تمارين

(أ) اكتب البرامج التي ترسم الأشكال الآتية :

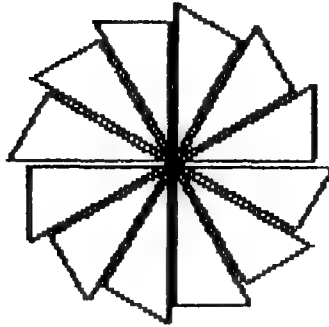


(ب) إذا أعطيت البرنامج التالي الذى يرسم الشكل الموضح بجواره :



FORWARD 50
LEFT 90
FORWARD 20
LEFT 120
FORWARD 30

فحاول رسم الشكل الآتى باستخدام البرنامج :



* الدوائر من المبادئ الأولية :

للكومبيوتر IBM مجموعة برامج خاصة يمكن استدعاؤها من القرص المغنطيسى
"LWIL procedures" وهذه البرامج تمكن من إعطاء أوامر جديدة للدوائر
والمحنيات مثل :

RCIRCLE n

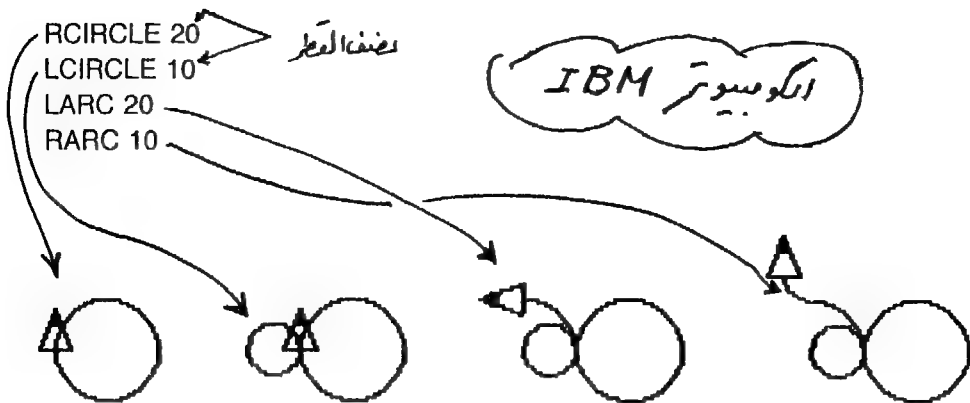
(لرسم دائرة على اليمين نصف قطرها n)

LCIRCLE n (لرسم دائرة على اليسار نصف قطرها n)
 RARC n (لرسم قوس على اليمين من دائرة نصف قطرها n)
 LARC n (لرسم قوس على اليسار من دائرة نصف قطرها n)
 ويتم استدعاء هذه الأوامر بإعطاء الأمر :

LOAD "CIRCLES

مع وضع القرص المغنطيسي LWIL في المكان المخصص له عند إعطاء الأمر والملاحق (١) والملاحق (٢) في نهاية الكتاب بهما شرح وإف لهذه البرامج كما يمكن نسخها وحفظها على القرص المغنطيسي لاستخدامها عند الحاجة إليها بنفس الأسماء مع أى كومبيوتر آخر .

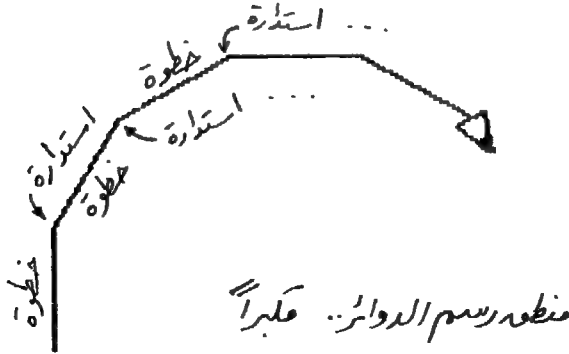
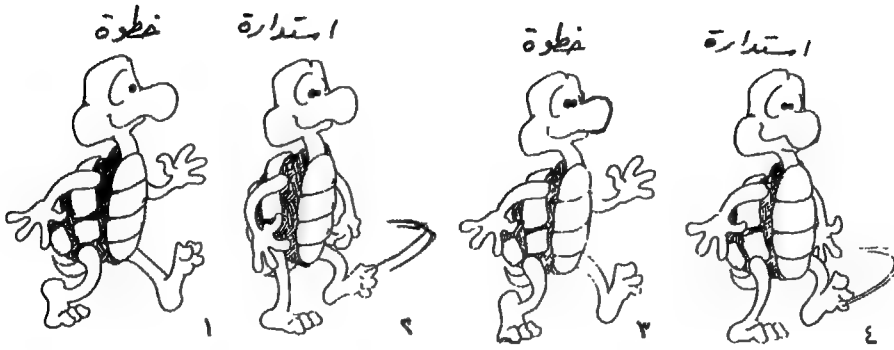
والمثال الآتى يوضح استخدام هذه الأوامر :



ومع ذلك فليس هذا هو موضوعنا :

فنحن نرغب في بناء البرنامج الذى يرسم دائرة باستخدام المبادئ التى تعلمناها . فما الفكرة في رسم الدائرة ؟ إن السلحفاة تتقدم خطوة وتستدير مرة وتتقدم خطوة أخرى وتستدير مرة ثانية وهكذا .. خطوة صغيرة يعقبها استدارة صغيرة .

وهذا ما تفعله السلحفاة في الشكل التالى وهى ترسم الدائرة .



بلغة لوجو يكون البرنامج كالآتي :

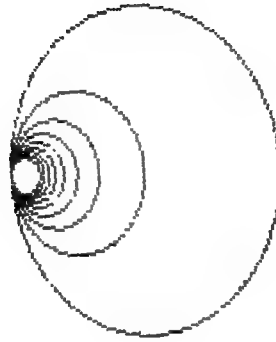
REPEAT 360 [FD 1 RT 1]

هذا يرسم دائرة على اليمين فإذا أردنا أن تكون الدائرة على اليسار نعدّل (RT) لتكون (LT) فيصبح البرنامج :

REPEAT 360 [FD 1 LT 1]

وبالتحكم في مقدار الدوران (RT) أو (LT) يمكن الحصول على أقطار مختلفة .

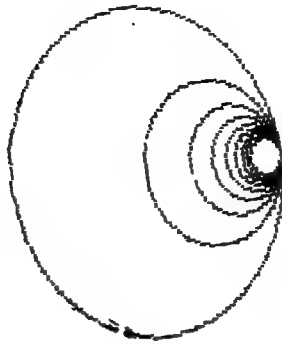
فالشكل الآتي بعد مرسوم بالبرنامج الأول (الدائرة على اليمين) . والدائرة الكبيرة تناظر (RT 1) أما الدوائر الأصغر فهي تناظر (RT 2) ، (RT 3) ، ... وهكذا حتى (RT 9) .



(منفذ على كوميونتر سنطير)

مجموعة دوائر على اليمين (RT)

وباستخدام الدوران اليسار LT بدلاً من RT نحصل على مرآة الصورة كما في الشكل التالي :



(منفذ على كوميونتر سنطير)

مجموعة دوائر على اليسار (LT)

كما يمكن جعل الدوائر تظهر أعلى أو أسفل صفحة الشاشة بإدارة السلحفاه مبدئياً قبل الرسم بمقدار ٩٠ درجة إلى أحد الجهتين اليسرى أو اليمنى .

والرسم التالي بعد يوضح مجموعة من الدوائر تم رسمها بعد إعطاء الأمر 90 LT وقد أظهرنا فيها السلحفاه حتى يظهر اتجاهها الأصلي الذي بدأت منه

وانتهت إليه . والرسم منفذ على الكمبيوتر سنكلير ويمكن تنفيذه على أى
كمبيوتر آخر ، لا فرق .



منفذ على
(كمبيوتر سنكلير)

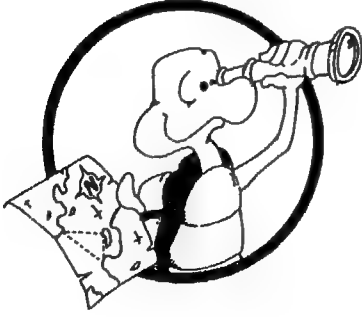
فلاش



إذا استخدمت الدوران RT 2 في
برنامج الدائرة فسوف تلاحظ أن السلفاه
ترسم الدائرة مرتين متطابقتين . وفي هذه
الحالة يكفي تكرار البرنامج بمقدار (360/2)
أى 180 مرة .

وكذلك بالنسبة للدوران (RT 3)
يمكن قسمة عدد مرات التكرار على 3 لتصبح 120 درجة وهكذا ..

تمرين



● ارسم مجموعة الدوائر التي تحتل النصف العلوي من الشاشة باستخدام درجات دوران مختلفة .

● اكتب البرنامج (الرئيسي) الذي يرسم مجموعتين من الدوائر على يمين ويسار الشاشة باستخدام البرامج الفرعية المناسبة .

فكرة



عرفنا تأثير درجة الدوران على نصف قطر الدائرة فكلما كبرت زاوية الدوران كان نصف القطر أقصر أو بمعنى آخر فإن السلحفاه تلف لفة سريعة وتعود إلى بيتها .

هل تستطيع أن تدرس بنفسك تأثير الخطوة FD على مساحة الدائرة ؟
إنها تجارب شائقة (ركز اهتمامك في نصف القطر ودرجة الدقة معاً) .

● ارسم ما تشاء من الأشكال باستخدام الدوائر :

وبتكرار برنامج الدائرة التكرار المناسب يمكن الحصول على ما نشاء من الأشكال الزخرفية التي تعتمد على الدائرة كأساس في تصميمها .
فالبرنامج التالي يرسم دائرة واحدة على يسار الشاشة :

```

TO CM ← اسم البرنامج
REPEAT 180 [ FD 1 LT 2 ]
END

```

لاحظ 360/2

وباستخدام هذا البرنامج كبرنامج فرعى يمكن رسم أشكال زخرفية مختلفة . والأمر الآتى يكرر الدائرة كل ٦٠ درجة (وبالتالى فقد استخدمنا أمر التكرار REPEAT 6) فنحصل على الشكل الموضح بعد :

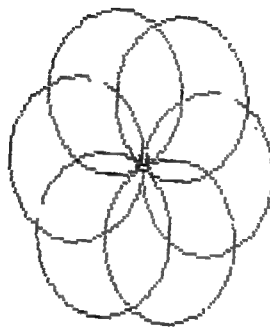
```

REPEAT 6 [ CM RT 60 ]

```

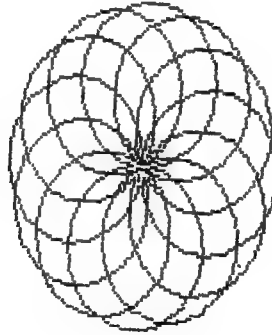
الدوران بعد كل دائرة

رسم الدائرة



رسم شكل زخرفى بالدوائر

أما البرنامج التالي فهو يكرر رسم الدائرة ١٢ مرة وبين كل دائرة وأخرى
تنحرف السلفاه بمقدار ٣٠ درجة . ونستخدم فيه البرنامج الفرعى السابق
: CM



TO CIRCLES

REPEAT 12 [CM RT 30]

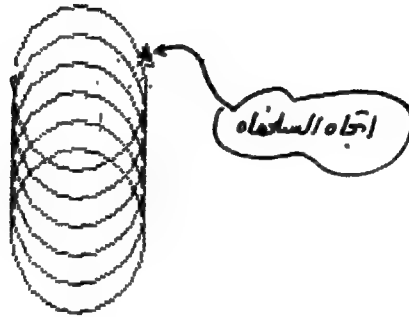
END

أما البرنامج الآتى فهو يرسم مجموعة من الدوائر الرأسية التى تعطى شكل
الاسطوانة المفرغة ويتم ذلك بالتحرك إلى أعلى عشر خطوات (FD 10) بعد
رسم كل دائرة مع رفع القلم (PU) ثم خفضه مرة أخرى (PD) والبدء فى رسم
الدائرة التالية :

TO CYLINDER

REPEAT 6 [CM PU FD 10 PD]

END

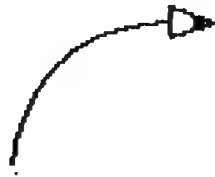


● المنحنيات والأقواس :

أمّا القوس فهو دائرة غير كاملة بمعنى أن عدد مرات التكرار أقل من أن يكمل الدائرة .

وهذا الأمر يرسم قوساً يمثل ربع دائرة :

REPEAT 90 [FD 1 RT 1]

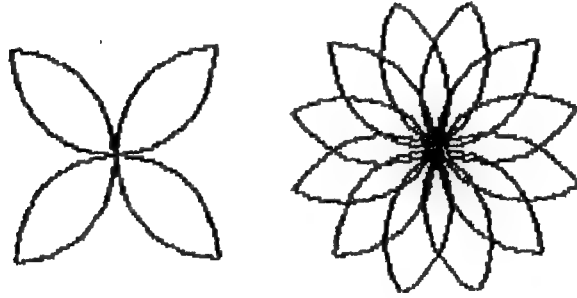


والأمر التالي يرسم قوساً يمثل نصف دائرة :

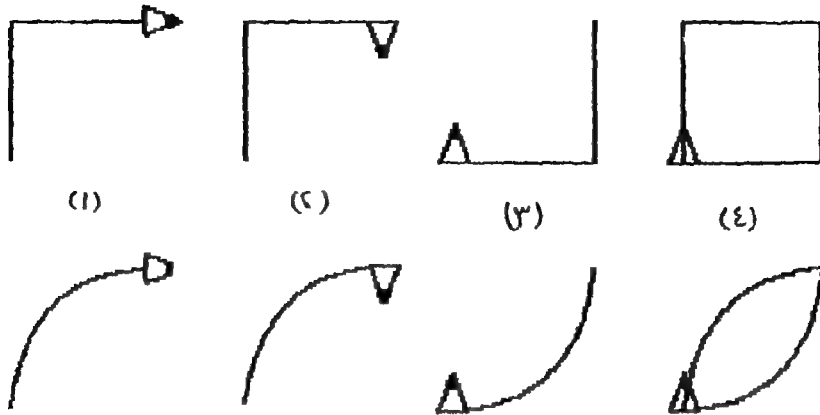
REPEAT 180 [FD 1 RT 1]



وبالتحكم في زاوية الدوران يمكن الحصول على أشكال مختلفة للمنحنيات سواء من ناحية جهة الدوران أو مقدار الدوران وهذه بعض الأشكال التي يمكن رسمها بالمنحنيات وهي تعتمد على وحدة رسم أصلية تشبه ورقة الشجرة أو بتلة الزهرة (PETAL) .



ورسم ورقة الشجرة يشبه كثيراً رسم المربع . تتبع الرسم التالي الذي يوضح خطوات رسم المربع والخطوات المناظرة لها لرسم ورقة الشجرة .



والبرنامج التالي يرسم ورقة الشجرة من المبادئ الأولية :

```

TO ARC1
REPEAT 45 [FD 1 RT 2]
RT 90
REPEAT 45 [FD 1 RT 2]
RT 90
END

```

* أما إذا كنت تعمل على الكمبيوتر IBM فيمكنك استخدام البرنامج التالي حيث نستخدم الأوامر الخاصة لرسم القوس مثل RARC :

```

TO PETAL
RARC 30
RIGHT 90
RARC 30
RIGHT 90
END

```

الكمبيوتر IBM



تنفيذ البرنامج

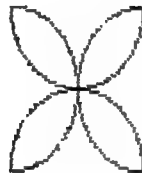
ويمكن استخدام أى من البرنامجين السابقين ARC1 أو PETAL كبرنامج فرعى بداخل البرنامج الرئيسى الآتى لرسم الشكل الموضح مع البرنامج :

```

TO ARC2
CS
REPEAT 4 [ARC1 RT 90]
END

```

اسم البرنامج الفرعى

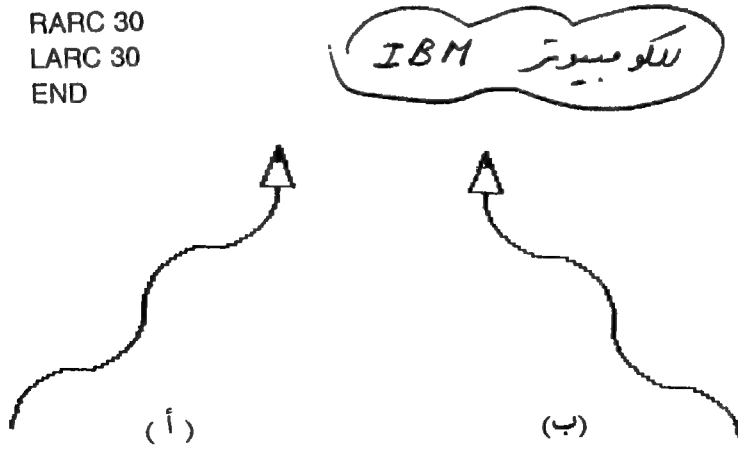


تنفذ على الكمبيوتر سنطير

● ارسم ثعباناً بالمنحنيات :

هذا البرنامج يرسم ثعباناً (على الكمبيوتر IBM) باستخدام خاصية القوس الأيمن RARC والقوس الأيسر LARC بالتتابع . ولتطويع البرنامج على أى كومبيوتر آخر يمكن بناء البرامج الفرعية المناظرة للأوامر LARC و RARC بالطرق التى تعلمناها أو استخدام هذه البرامج وهى فى الملحق (١) .

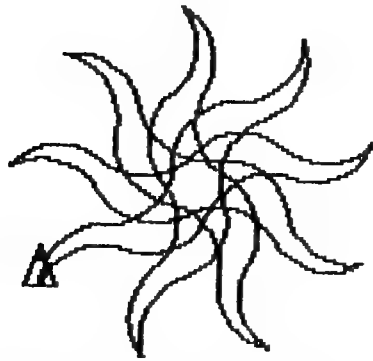
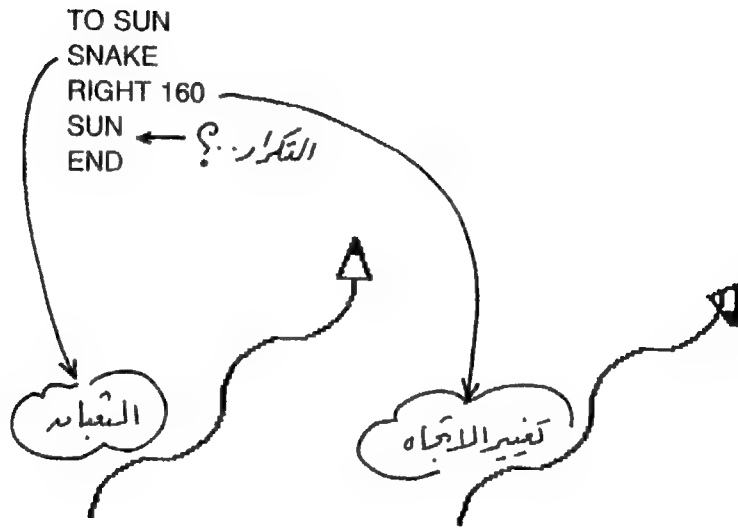
```
TO SNAKE  
RARC 30  
LARC 30  
RARC 30  
LARC 30  
END
```



الثعبان وصورته

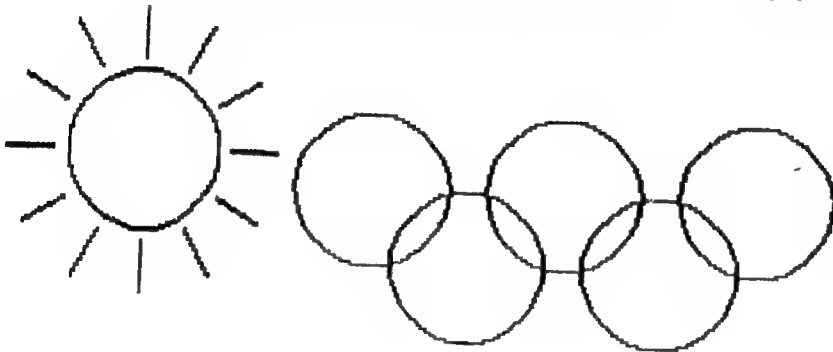
والشكل يوضح الثعبان (أ) الناتج من البرنامج وصورته (ب) التى يمكن برمجتها بتبديل الانحرافات إلى اليمين وإلى اليسار .

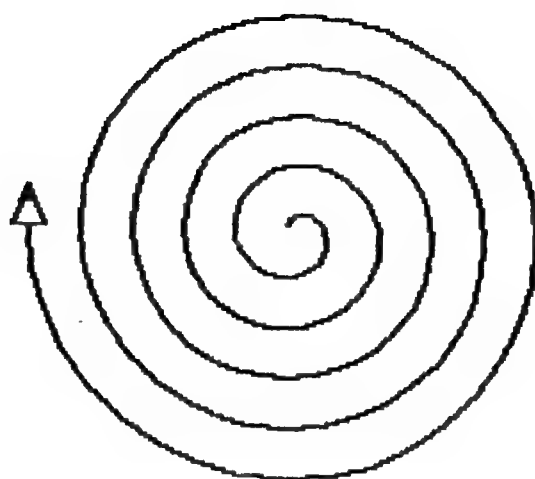
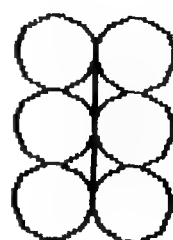
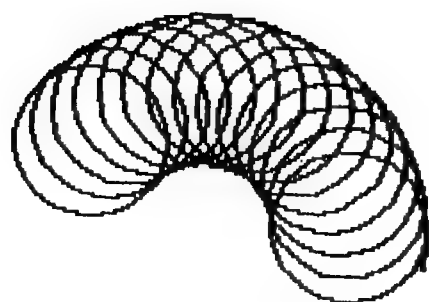
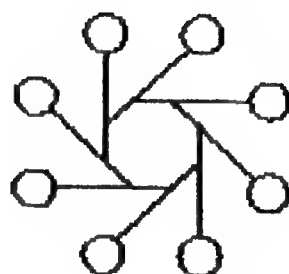
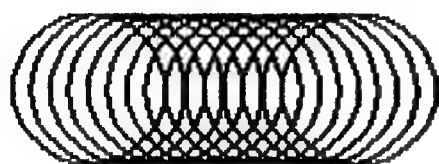
ويمكن استغلال هذا البرنامج لرسم الشمس الموضحة فى الشكل التالى باستخدامه كبرنامج فرعى . وسوف يتوقف بعض القراء أقوىاء الملاحظة عند الأمر الأخير فى البرنامج الذى يحمل نفس اسم البرنامج SUN .. هل هذا ممكن ؟ إن هذه الطريقة تتميز بها لغة لوجو للتكرار وسوف نتحدث عنها تفصيلاً فى الأبواب القادمة .



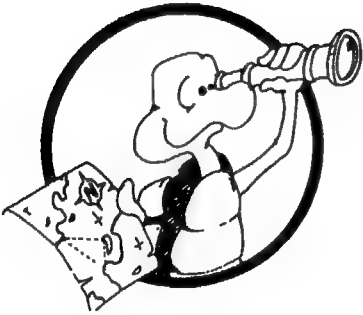
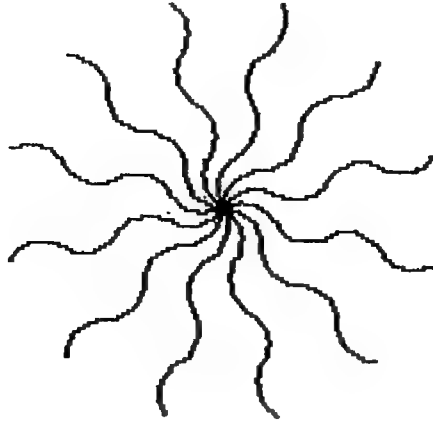
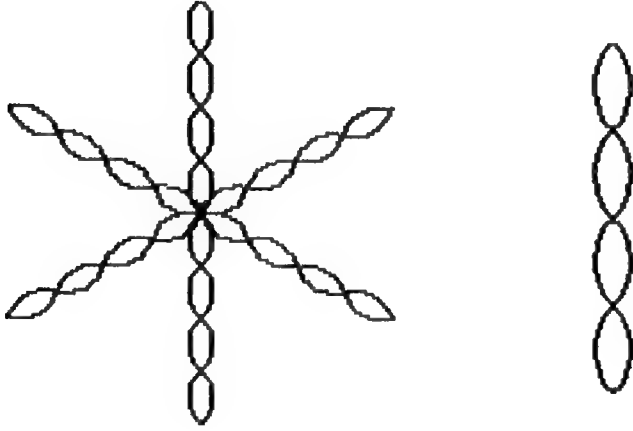
مع التكرار
نحصل على الشمس

تمارين (أ) حاول رسم الأشكال الآتية باستخدام الدوائر :



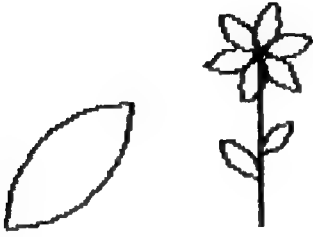


(ب) وحاول في الأشكال الآتية باستخدام المنحنيات :



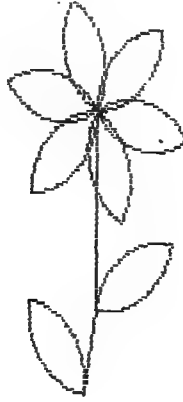
● تطبيقات :

١ حدائق الزهور :



عرفنا في الأبواب السابقة كيف نرسم ورقة الشجرة أو بتلة الزهرة، ومن البتلات يمكن تجميع زهرة كاملة باستخدام وسائل التكرار المناسبة .

والبرنامج التالي ARC3 يرسم غصن الشجرة الموضح معه باستخدام البرنامج الفرعي ARC1 الذي يرسم بتلة واحدة (أو ورقة شجر .. لا فرق بالنسبة للكمبيوتر) .



البرنامج

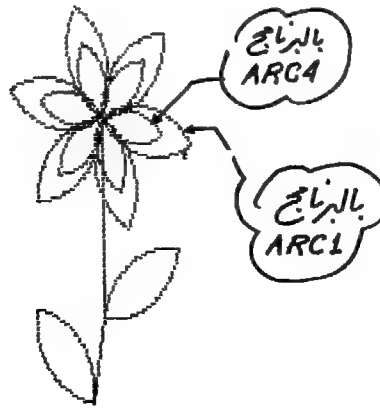
```
TO ARC3
CS HT FD 30
REPEAT 6 [ARC1 LT 60]
BK 70 RT 10 ARC1
BK 30 LT 90 ARC1
END
```

البرنامج الفرعي

```
TO ARC1
REPEAT 45 [FD 1 RT 2]
RT 90
REPEAT 45 [FD 1 RT 2]
RT 90
END
```

وسواء كنت تستخدم البرامج الجاهزة للأقواس والدوائر أو كنت تصممها بنفسك فإنه يمكن التحكم في مساحة الورق أو البتلة بالتحكم في زاوية انحناء القوس .

والبرنامج الآتي بعد هو تطوير للبرنامج السابق حيث يرسم الزهرة المركبة باستخدام برنامج فرعي جديد ARC4 علاوة على البرنامج الفرعي السابق ARC1 أما اسم البرنامج الرئيسي فهو FLOWER .



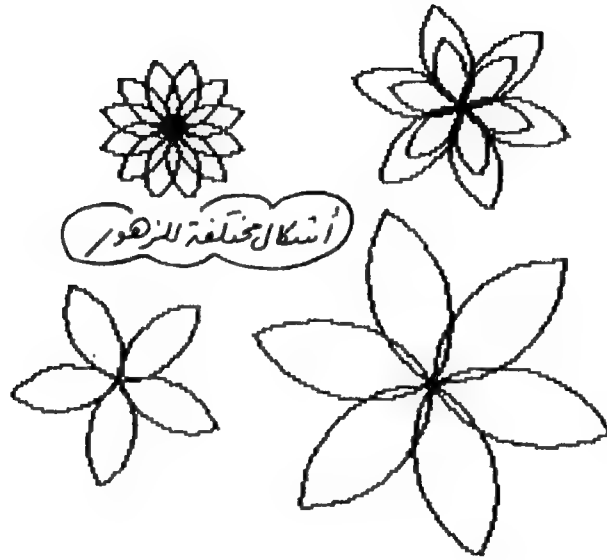
البرنامج

```
TO FLOWER
CS HT FD 30
REPEAT 6 [ARC1 ARC4 LT 60]
BK 70 RT 10 ARC1
BK 30 LT 90 ARC1
END
```

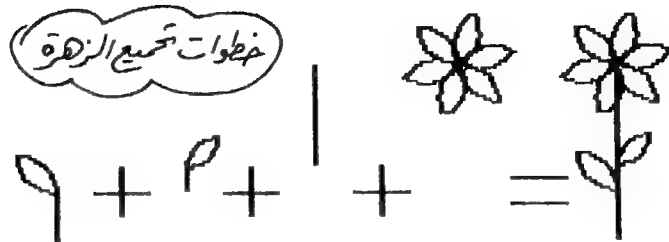
البرنامج الفرعي ARC4

```
TO ARC4
REPEAT 90/3 [FD 1 RT 3]
RT 90
REPEAT 90/3 [FD 1 RT 3]
RT 90
END
```

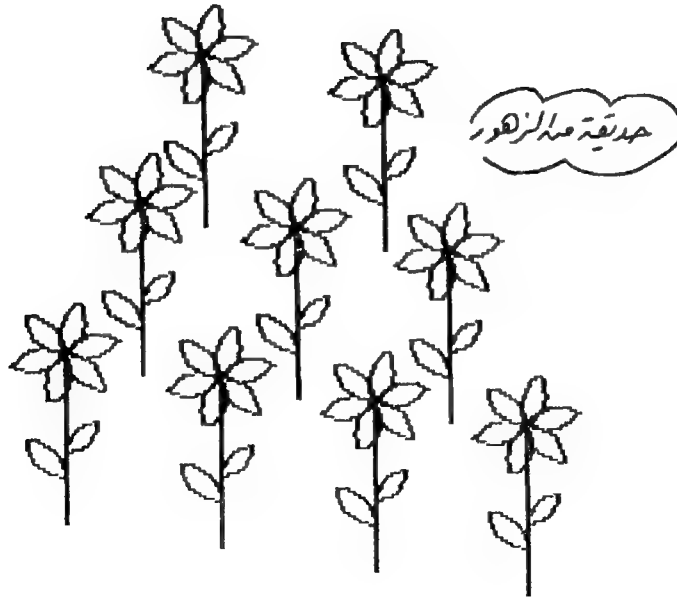
ولا نهاية للأشكال التي يمكن تصميمها باستخدام المنحنيات من الزهور والأوراق والتي تستخدم كوحدة بناء لمجموعات متكاملة من الزهور .



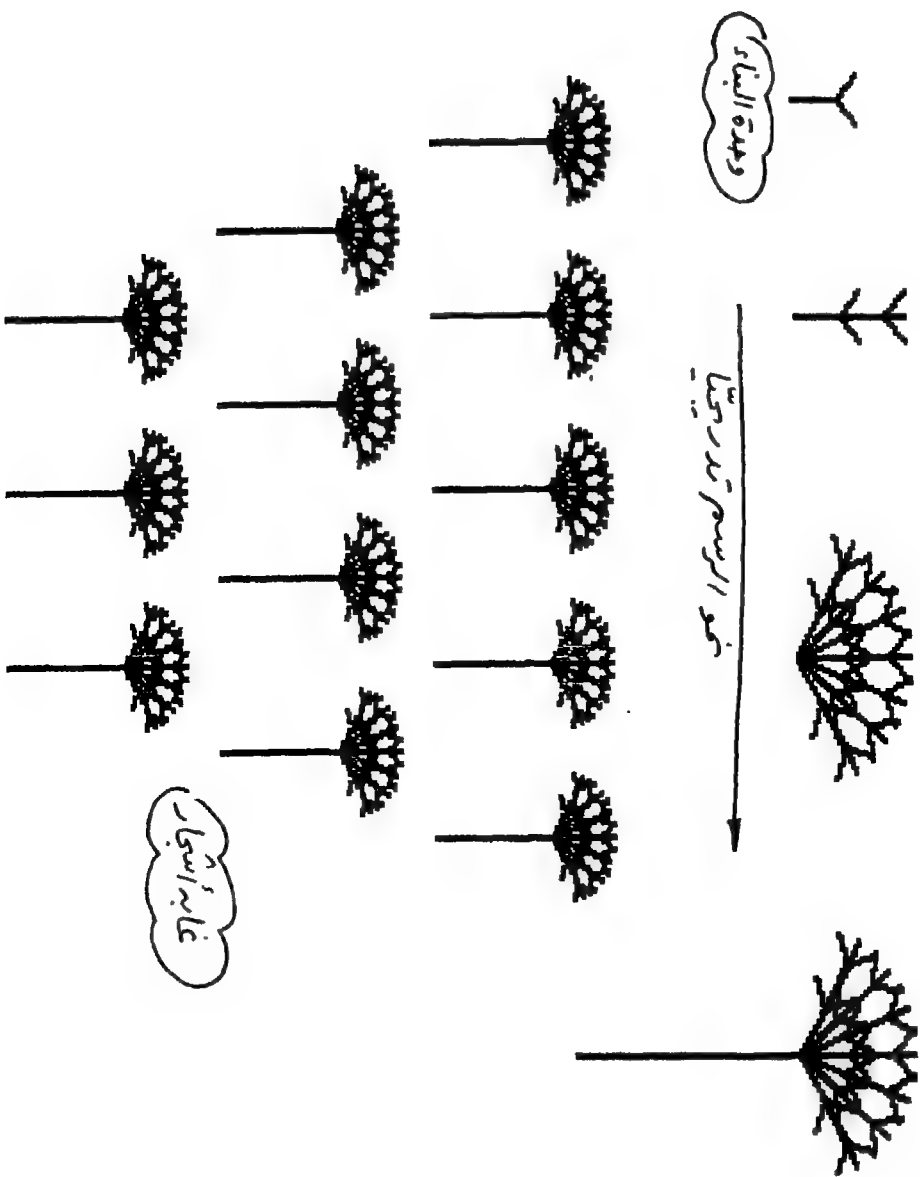
ومن الممكن تقسيم الزهرة إلى عدة أجزاء حتى يسهل رسمها . فمثلاً الزهرة الموضحة يمكن تجميعها من الأشكال المجاورة لها والمكونة من الزهرة السداسية التي رسمناها من قبل والساق (الخط المستقيم) والورقة المائلة إلى اليمين والأخرى المائلة إلى اليسار وهذه هي فكرة البرنامج السابق .



وبناء أول زهرة يمكن تكرار الرسم على مسافات عشوائية أو منتظمة للحصول على الحديقة كاملة .



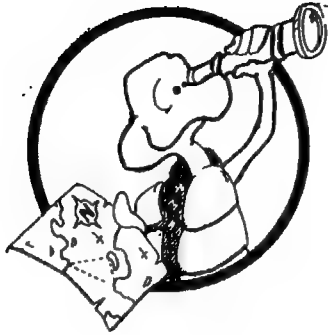
أما الشكل التالي فيوضح كيفية بناء غابة من الأشجار الكثيفة باستخدام وحدة بناء صغيرة لا تحتاج إلى أي مجهود في البرمجة :

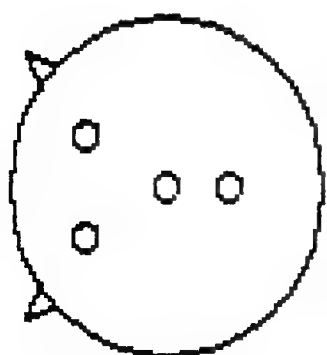
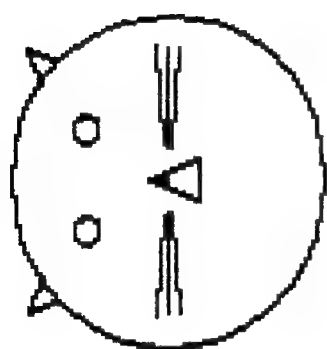


٢ الملامح والوجوه :

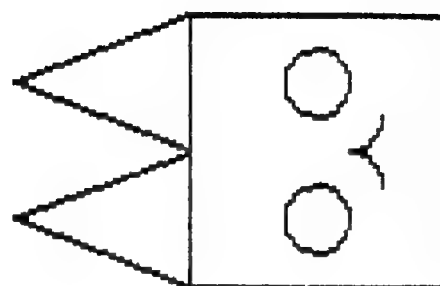
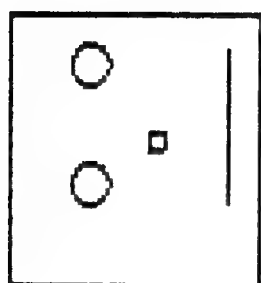
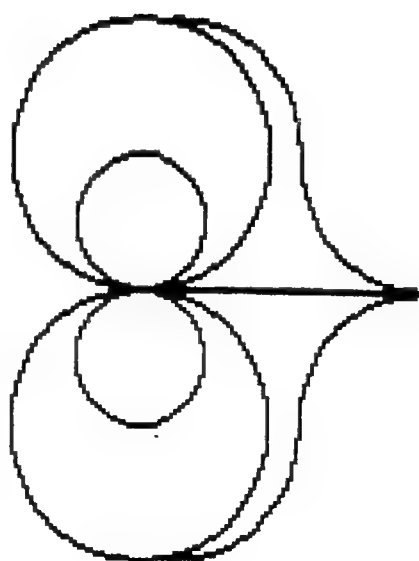
فيما يلي لن نتعرض لتفصيلات البرامج ولكننا نعرض بعض الأفكار عن تكوين الملامح والوجوه بأبسط الطرق فالأشكال الآتية بعد هي أشكال بسيطة نستخدم فيها الدوائر أو المربعات والدوائر معاً . وأهم ما يميّزها التماثل . وفي هذه الحالة تفيدنا خاصية « الصورة على المرآة » في إتمام الرسم بمجهود أقل في البرمجة . كما نلاحظ أن السلحفاة قد ظهرت في أحد الأشكال لترسم شكل أنف القطرة ذات الشارب .

حاول رسم هذه الأشكال كتدريب على استخدام الدوائر والمربعات .

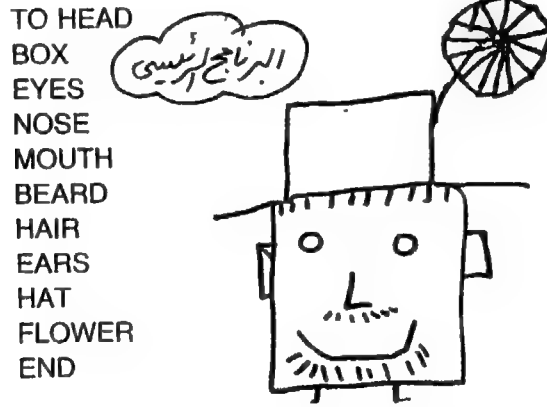




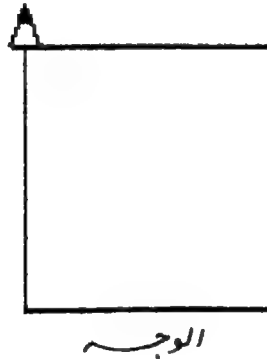
تشكيلة من الوجوه

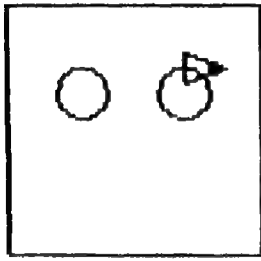


وهذه فكرة برنامج متكامل لرسم الوجه البشرى الموضح فى « الاسكتش »
التالى :

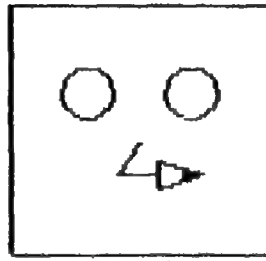


وكما نرى أنه يمكن تقسيم البرنامج الرئيسى إلى عدة أجزاء أو برامج فرعية .
فالوجه نعبر عنه بالمربع (البرنامج BOX) والعينان يتم رسمهما بالدوائر بالبرنامج (EYES) أما الأنف فهي خطّان (NOSE) والفم عبارة عن قوس واحد ، لأعلى إذا كان الرجل مبتسماً ولأسفل إذا كان معكراً المزاج (MOUTH) .
يلى ذلك اللحية (BEARD) والشعر (HAIR) وهما خطوط مستقيمة والأذنان وهما مربعان صغيران (EARS) ثم القبعة (HAT) وهى مربع أيضاً ، والوردة (FLOWER) وهى عبارة عن قوس ينتهى إلى شكل من الأشكال التى رسمناها من قبل .





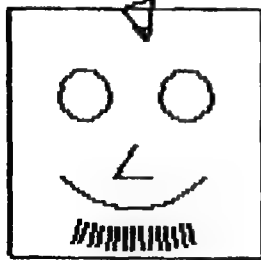
العيان



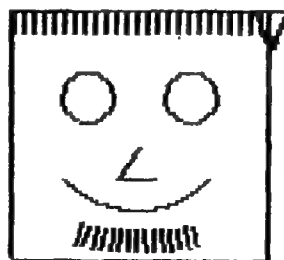
الأنف



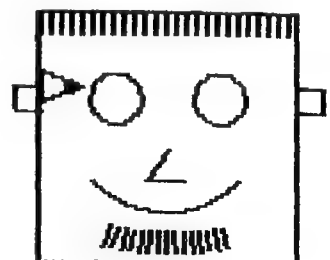
الفم



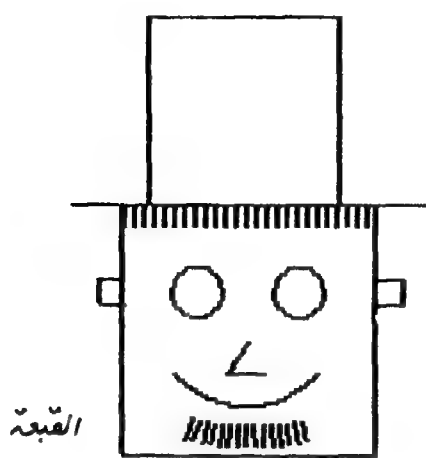
الحيمة



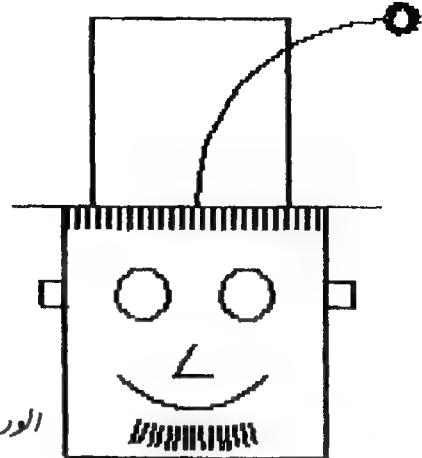
الشعر



الأذنان



القبعة



الوردة

٣ حيوانات لوجو :

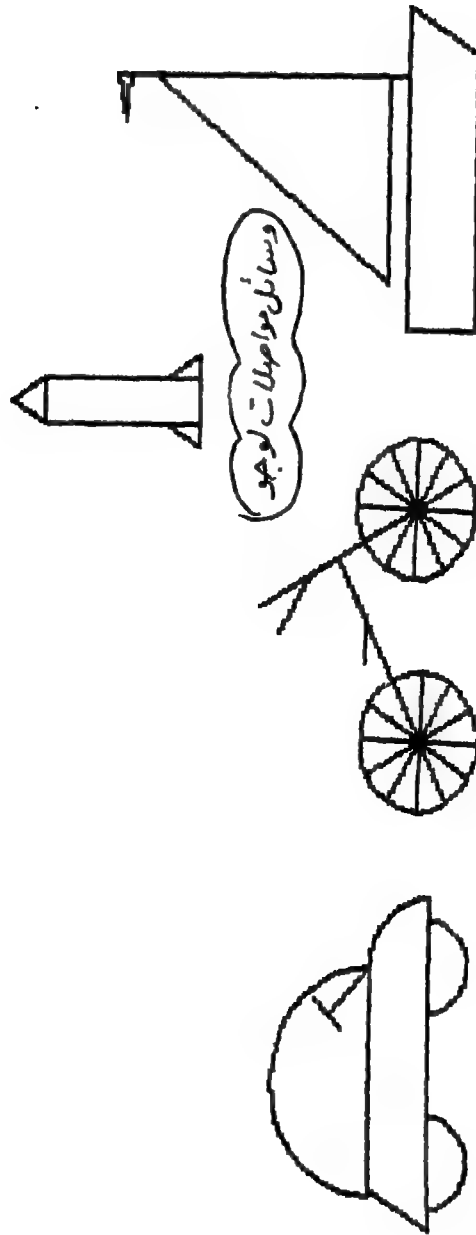
وباستخدام وحدات بناء بسيطة أيضاً يمكن رسم نوعيات مختلفة من الحيوانات الموضحة في الشكل التالي بعد . فالأسماك عبارة عن أزواج من الأقواس المتقاطعة . والفراشة تبني من الخطوط والمثلثات . أما القطة والفأر

فئينان من الدوائر والأقواس بصفة أساسية .



٤ وسائل المواصلات أيضاً :

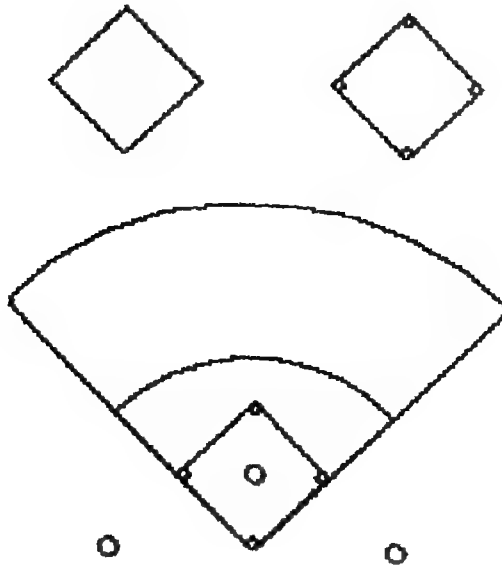
وهذه بعض نماذج من وسائل المواصلات يمكن تصميمها بأبسط الطرق وتشمل المركب الشراعية والسيارة والعجلة والصاروخ .

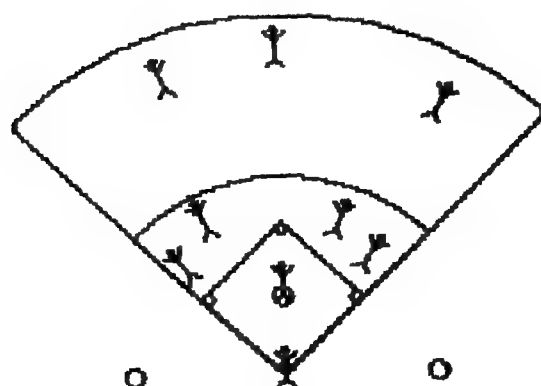
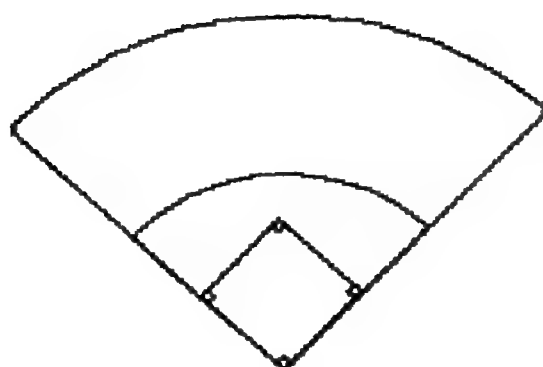


٥ مشروع لعبة كومبيوترية :

وهذه خطوات لرسم ملعب « البيسبول » حيث يبدأ المشروع بمربع واحد ثم ينتهى إلى شكل الملعب المتكامل وفيه اللاعبون .

ولعل بعض الأصدقاء يستطيعون التحول من المشروع من مجرد الرسم إلى بناء اللعبة الكومبيوترية الكاملة بعد التمكن من لغة لوجو .





٥

عالم الأرقام والحروف

TEXTSCREEN

● شاشة الكتابة

تعرضنا بسرعة من قبل للأمر TEXTSCREEN أو بالصورة المختصرة TS وعلمنا أن هذا الأمر يسمح ما على الشاشة من كتابة أو رسم ويضع علامة الاستعداد (؟) يسار أعلى الشاشة ، وهو مكان مناسب لها عندما نريد أن نكتب أو نحسب بعض الحسابات بلغة لوجو .



● البيانات في لغة لوجو :

البيانات التي يتعامل معها الكمبيوتر من خلال لغة لوجو ثلاثة أنواع :

١ — الأعداد (Numbers)

وهو يستطيع مع الأعداد أن يؤدي مختلف العمليات الحسابية والمنطقية كما سنرى .

٢ — الكلمات (WORDS)

أما الكلمات فهي التراكيب المختلفة لعناصر اللغة مثل الحروف الأبجدية والأرقام والعلامات الخاصة .

٣ — القوائم (LISTS)

والقوائم هي تركيبة من مجموعة كلمات .

وسوف نرى في الفقرة القادمة كيف يتعامل الكمبيوتر مع كل نوع من أنواع هذه البيانات .

(أ)

●● التعامل مع الأعداد في لغة لوجو ●●

● العمليات الحاسوبية :

الأرقام هي أحب شيء إلى الكمبيوتر . لأنه يرى العالم الخارجى فى شكل أرقام . ولو أنك أعطيتة بياناً ما مكوناً من الحروف الأبجدية مثل اسم شخص ، فإنه فى الحقيقة يخزن فى صورة أرقام . لكنك عندما تطلب منه طبعه على الشاشة فإنه يحوله مرة أخرى إلى حروف حتى نفهم نحن !
والآن نرى معاً كيف يمكن إجراء بعض الحسابات على الكمبيوتر باستخدام لغة لوجو ولنبدأ بعملية الجمع :

أمر الجمع ← `PRINT 4 + 5` ?

النتيجة ← 9

علامة الاستعداد ← ?

إذن فعلمية الجمع تتم باستخدام الأمر `PRINT` . والرمز المستخدم فى عملية الجمع هو (+) . ويراعى ترك مسافة بين الكلمة `PRINT` والأرقام التى تليها .

كذلك فإن الرمز المستخدم فى عملية الطرح هو (-) وفى الضرب هو (*) وفى القسمة هو (/) .

ولنجرّب الأمثلة الآتية :

`PRINT 4 + 5`

`PRINT 36 - 6`

`PRINT 36 / 7`

`PRINT (3 + 5) * 7`

وفى المثال الأخير نلاحظ ظهور الأقواس مع الأعداد . لماذا ؟

إن هذا القوس يخبر الكمبيوتر أن يجمع العددين 3 + 5 أولاً ثم يضرب

الناتج في العدد 7 .

ولو كانت العملية الأولى عملية طرح لزم استخدام الأقواس أيضاً . ولكن إذا كانت العملية الأولى عملية ضرب أو قسمة فيمكن الاستغناء عن الأقواس لأن الكمبيوتر ينفذ الضرب والقسمة قبل الجمع والطرح عادة .

وللتأكد من ذلك جرب الآتي :

PRINT 3 + 5 * 7

PRINT (3 + 5) * 7

الناتج الذي نحصل عليه من العملية الأولى هو 38 .

والناتج الذي نحصل عليه من العملية الثانية هو 56 .

أى أن الكمبيوتر في العملية الأولى قد أنجز عملية الضرب قبل عملية الجمع وفي العملية الثانية أنجز عملية الجمع قبل عملية الضرب .

TRUE, FALSE

● المقارنات

يمكن للكمبيوتر بجانب العملية الحسابية أن يتخذ قراراً منطقياً أيضاً . والقرار المنطقي يؤدي إلى أحد نتيجتين : « صحيح » (TRUE) أو « خطأ » (FALSE) .

والقرار المنطقي يكون عادة إجابة لسؤال نسأله نحن للكمبيوتر مثل :

PRINT 5 = 8

هذا معناه : « هل الرقم 5 يساوى الرقم 8 » ؟

ولأن الإجابة هي « لا » فإن الكمبيوتر يطبع على الشاشة كلمة FALSE . فلنسأله الآن هذا السؤال :

PRINT 5 < 8

هذا معناه : « هل 5 أصغر من 8 » ؟

الإجابة هي TRUE بمعنى نعم .

جرب الآن السؤال الآتي :

PRINT 5 > 8

وهذا معناه : « هل 5 أكبر من 8 » ؟
الإجابة المنتظرة هي FALSE بمعنى لا .
ومن الجائز أيضاً أن يشمل السؤال عملية حسابية مع القرار المنطقي المطلوب مثل :

PRINT 5 = (8 - 3)

وهذا معناه : « هل 5 مساو لناتج عملية الطرح (8 - 3) » ؟
والإجابة هي TRUE .

فلاش

هل نحتاج إلى الأقواس في المثال السابق ؟
في الحقيقة لا . لأن الكمبيوتر يؤدي أية عملية حسابية قبل المقارنات . لذلك فهو يحسب أولاً المقدار (8 - 3) بصرف النظر عن وجود الأقواس ثم يقارن العدد 5 بالنتيجة .
وجرب الآتي للتأكد :

PRINT 5 = 8 - 3.

PRINT 5 = (8 - 3)



● مثال :

هذه بعض العمليات الحسابية على الكمبيوتر المنزلى سنكلير نقدمها كمثال عام لما سبق . ولعلنا نلاحظ أن لغة لوجو للكمبيوتر IBM هي للكمبيوتر سنكلير أو غيره ، فى حدود التطبيقات التى نعرضها فى هذا الكتاب ومع ذلك فسوف نشير إلى الأجزاء التى قد يختص بها كومبيوتر IBM إذا تعرضنا لها :

ونلاحظ فى هذا المثال استخدام الأمر المختصر PR بدلاً من PRINT .

كومبيوتر سنكلير

```

WELCOME TO SINCLAIR LOGO
© SOLI/LCSI 1984 VER. 1.6
?pr 5+8
13
?pr 5*5
25
?pr 55>6
TRUE
?pr 6<5
FALSE
?pr 6=26-20
TRUE
?pr 6=(26-20)
TRUE
?
```

} عمليات حسابية

} مقارنات

● هناك دائماً طريقة أخرى :

يمكن أيضاً أداء العمليات الحسابية بطريقة مختلفة بدون استخدام المؤثرات (+ * /) وهذه بعض أمثلة :

* عملية الضرب :

PR PRODUCT 4 4 ← الأمر
16 ← الإجابة

* عملية القسمة :

PR DIV 7 4
3.5

* عملية الجمع :

PR SUM 50 40
90

● عملية المقارنة (=) :

PR EQUALP 5 7
FALSE

الأمر
الإجابة

ملاحظة

هل لاحظت ظهر العلامة العشرية (.) في مثال القسمة . إن لوجو تتعامل مع الأعداد الصحيحة والأعداد الحقيقية (التي تحتوى على كسور) وفي الفقرة القادمة نقدم بعض وسائل التعامل مع الكسور بالتقريب أو الحذف .

(ROUND)

● التقريب

اكتب الأمر التالى على الشاشة وشاهد النتيجة :

PRINT ROUND 5.463

نتيجة هذا الأمر هى طبع العدد 5 .

جرب الآن الأمر التالى :

PRINT ROUND 5.964

نتيجة هذا الأمر هي العدد 6 .

أى أن عملية التقريب قد أجريت على الكسر لأقرب رقم صحيح .
وهذه هي إجابات الكمبيوتر سنكثير :

```
?print round 5.463
5
?print round 5.964
6
```

INT

● الحذف

أما هذه العملية فينتج عنها حذف الجزء الكسرى من العدد الحقيقى بصرف النظر عن قيمة الكسر .
أنظر هذه الأمثلة :

```
? PR INT 5.2129
```

5

```
? PR INT 5.5
```

5

```
? PR INT -5.5
```

-5

RANDOM

● الأعداد العشوائية

أما العملية RANDOM فهي تشبه النرد (زهر الطاولة) بمعنى أنها تنتج أى رقم عشوائى تماماً كما تلقى النرد فتظهر على سطحه المواجه لك أية أرقام عشوائية .

وحدود الرقم العشوائى يتحكم فيها العدد المُدخل مع كلمة RANDOM كالمثال الآتى :

? PR RANDOM 6

هذا الأمر يطبع عدداً عشوائياً يتراوح بين صفر وخمسة .
وأيضاً :

? PRINT RANDOM 50

هذا الأمر يطبع على الشاشة عدداً عشوائياً بين صفر و ٤٩ .
أى أن الحد الأقصى للعدد الذى تنتجه كلمة RANDOM أقل من الرقم
المُدخل بعدها بمقدار 1 .

لذلك يمكن كتابة الكلمة RANDOM فى صورة عامة كالآتى :

RANDOM n

ويكون أقصى عدد تنتجه كلمة RANDOM هو $(n - 1)$.

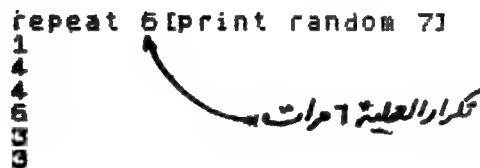
● مثال :

يمكن هنا استخدام أمر التكرار REPEAT لإنتاج عدد من الأرقام العشوائية
التي نستطيع استغلالها فيما بعد. فى الكثير من التطبيقات لا سيما فى الرسم .

وهذا المثال منفذ على الكمبيوتر المنزلى سنكلىر :

```
repeat 6[print random 7]
```

تكرار العملية ٦ مرات



الأرقام المطبوعة هنا تتراوح بين الصفر والستة .

تجربة

هل تستطيع إجراء تعديل على
المثال السابق بحيث تحاكي النتيجة
عملية رمى النرد أى تطبع أرقاماً
تتراوح بين الواحد والستة .

● عالم الأعداد فى لغة لوجو :

إن عالم الأعداد فى لغة لوجو عالم متسع . وهى تقدم إمكانيات كثيرة جداً
للتعامل مع الأعداد ربما لا توجد بها لغة أخرى من لغات الكمبيوتر .

ولا نعتقد أن مستوى هذا الكتاب يكفى لأن نقول كل شئ عن إمكانيات
التعامل مع الأعداد ، فبعض هذه الإمكانيات لا يهم إلا الرياضيين ومع ذلك
فاعتقادي أن إشارة سريعة لمثل هذه التطبيقات تكفى لإرشادهم إلى الطريق .

وهذه بعض العمليات الحسابية والرياضية التى تتضمنها لغة لوجو :



● النسب المثلثية :

تحتوى لغة لوجو على النسب المثلثية الآتية :

$\left\{ \begin{array}{l} \text{SINE } n \\ \text{SIN } n \end{array} \right\}$
حا

$\left\{ \begin{array}{l} \text{COSINE } n \\ \text{COS } n \end{array} \right\}$
جنا

$\left\{ \begin{array}{l} \text{TANGENT } n \\ \text{TAN } n \end{array} \right\}$
ظا

● مثال :

? PR TAN 50
 1.1917536

● النسب المثلثية العكسية :

ARCSIN n جا-١

ARCCOS n جنا-١

ARCTAN n ظا-١

● مثال :

? PR ARCTAN 1 ← الأمر
 45 ← الإجابة

● مقلوبات النسب المثلثية :

$\left\{ \begin{array}{l} \text{COTANGENT } n \\ \text{COT } n \end{array} \right\}$
ظنا

● مثال :

? PR COT 45 ← الأمر
 1 ← الإجابة

● باقى القسمة

تعطى هذه العملية باقى قسمة العدد a على العدد b كلاً مثله الآتية :

اطبع باقى قسمة ١٦ على ٤ ? PR REMAINDER 16 4
 الإجابة .. صفراً 0 ←
 ? PR REMAINDER 16 5
 1

وفى هذا المثال نرى أن باقى قسمة 16 على 5 هو العدد 1 .

● إيجاد الجذر التربيعى SQR T n
 ● مثال :

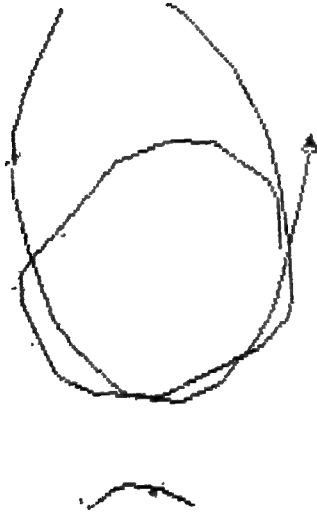
? PR SQR T 49
 7

? PR SQR T 4567
 67.596

● مثال عام الرسم فى إحداثيات عشوائية :
 لو أنك أدخلت هذا الأمر إلى الكمبيوتر :

? REPEAT 50 [FD RANDOM 30 LT RANDOM 30]

فسوف تشاهد السلحفاه تتحرك فى اتجاهات عشوائية فتحصل على
 رسومات عشوائية على الشاشة بمعنى أنها يختلف شكلها فى كل مرة يُنفذ فيها
 هذا الأمر . والشكل التالى هو أحد
 الأشكال العشوائية التى يمكن
 مشاهدتها على الشاشة .



أحد أشكال
التنفيذ العشوائية للرسم

(ب)

●● التعامل مع الكلمات والقوائم في لغة لوجو ●●

LOGO WORDS

● ما هي كلمات لوجو ؟

تتكون أى لغة من الحروف والأرقام والعلامات الخاصة الموجودة على لوحة الأزرار . وهناك اسم عام نطلقه على أى من هذه المكونات وهو « اللبنة » (character) واللبنة معناها أصغر جزء في بناء اللغة (أو الحجر الذى يُبنى منه المنزل) .

والكلمة في لغة لوجو هي تركيبة من هذه اللبنات ولا يشترط أن تحمل معنى ما . وعادة تسبق كلمة لوجو علامة الاقتباس المزدوجة (") (ولكنها لا توضع في مؤخرتها مثل اللغات الأخرى) .

ومع ذلك فهناك بعض اللبنيات لا يجوز على أن تحتوى عليها الكلمة مثل :

+ , - , * , / , (,) , [,]

ولنتجرب معاً هذه الأوامر ونشاهد رد فعل لوجو :

```
PRINT "HELLO
PRINT "ABCXYZ
PRINT "R2D2
PRINT "AB.$ -) * *
PRINT "3 + 4
```

وتعتبر الأرقام أيضاً كلمات (WORDS) في لغة لوجو وقد تستخدم مع علامة الاقتباس أو بدونها كالمثال الآتي :

```
PRINT "25
PRINT "25 + "25
```

في الأمر الأول يطبع الكمبيوتر العدد 25 .

وفي الأمر الثاني يطبع العدد 50 كمجموع لكل من 25 ، 25 .

وهذا يكافئ تماماً الأمر :
PRINT 25 + 25

ملاحظة



هناك فرق بين الحرفيات (strings) في لغة بيسك وبين الكلمات في لغة لوجو ولذلك يجب الحذر من الخلط بينهما بالنسبة لمن تعلم لغة بيسك من قبل .

ويجوز أن تكتب العلامة (") بعد الأمر PRINT دون أن يليها شيء ما .
وهذا بمثابة الأمر « اترك سطرًا خاليًا » أى :

? PRINT

● تحديد نهاية الكلمة :

وتنتهى كلمة لوجو عند أول مسافة خالية . فالكلمة لا يجوز أن تحتوى على
مسافة خالية ، تماماً مثل كلمات اللغة العربية أو الإنجليزية .
أدخل هذا الأمر للكمبيوتر وشاهد الرد على الشاشة :

? PRINT " my name is SALLY

في هذه الحالة سوف يعتبر الكمبيوتر أن (my) هى الكلمة المقصودة أما ما
بعدها فسوف يشكو من عدم الفهم له لذلك يكون رد الفعل كالاتى :

الأمر	←	? PRINT " my name is SALLY
	←	my
الإجابة		
الشكوى	←	I don't Know how to NAME
	←	?
علامة استعداد جديدة		

إذن فلنجرّب إعطاء هذا الأمر الجديد حيث نضع علامة اقتباس أمام كل
كلمة :

الأمر ← PRINT "my " name " is "SALLY ?

الإجابة ← my

You don't say what to do with name

?

شكوى جديدة

آه لقد عرف أن الكلمة التالية لكلمة my هي أيضاً كلمة ولكنه توقف عندها أيضاً وأرسل لنا رسالة بالخطأ تحمل الشكوى : « لم تخبرني ماذا أفعل بكلمة name » .

وبالطبع فإنه لم يقرأ بقية الجملة . هناك حل وسوف نعرفه في حيته .

LISTS

● القوائم في لغة لوجو

أما القوائم فهي تركيبة من كلمات لوجو .

ونضع القائمة عادة بين قوسين مربعين [] ولنكتب هذه الأمثلة ونشاهد النتيجة على الشاشة :

```
PRINT [HELLO THERE]
```

```
PRINT [1 2 3 4 5 6]
```

```
PRINT [MY NAME IS DAN]
```

```
PRINT [THIS IS A LIST: [THIS IS A LIST: ]]
```

```
PRINT [[A B] [C D] [E F]]
```

ولعلنا نلاحظ أنه بداخل القوسين المربعين يمكننا أن نستخدم أى لبنة من اللبنات الموجودة على لوحة الأزرار حتى لو كان القوس المربع نفسه . فأمر الطباعة هنا ينسخ كل ما بداخل القوسين ويطبعه على الشاشة . والقوائم في لغة لوجو تناظر الحرفيات strings في لغة بيثون .

ويجوز أن نترك القوسين المربعين فارغين وهذا معناه ترك سطر خال أيضاً .

```
PRINT [ ]
```

WORD

● توصيل كلمتين معاً

هناك أوامر خاصة بالكلمات والقوائم يمكننا من خلالها أن نكون التراكيب المختلفة فمثلاً الأمر WORD يقوم بوصل كلمتين معاً كالاتي :

```
PRINT WORD "BIG "WORD
PRINT WORD "WO "RD
```

في المثال الأول يقوم الكمبيوتر بطبع كلمة BIGWORD .
وفي المثال الثاني يقوم بطبع كلمة WORD .

ولكن لا يجوز توصيل أكثر من كلمتين بهذه الطريقة ، ولو أنك حاولت فسوف يتوقف الكمبيوتر عند الكلمة الثالثة ويرسل لك الشكوى المعروفة :
« ماذا أفعل بهذه الكلمة الثالثة .. » .

● توصيل أكثر من كلمتين (WORD "...

يمكن للأمر WORD أن يقبل أكثر من كلمتين بشرط استخدام الأقواس العادية () لتشمل كل الكلمات المدخلة كالمثال الآتي :

```
PRINT (WORD "BIG "GER "WORD )
```

↑ مسافة خالية

هنا سيقوم الكمبيوتر بتوصيل كل ما بين القوسين ويطبع على الشاشة الكلمة BIGGERWORD .

ملاحظة



هناك مصادر شهيرة للخطأ في الاستخدام السابق للأمر WORD أولها : ضرورة ترك مسافة خالية قبل القوس الأخير "()" وإلا سيكتب الكمبيوتر هذا القوس ضمن الكلمة الجديدة ظناً منه أنه جزء من الكلمة الأخيرة (في أغلب الطرازات) .

أما الخطأ الشائع الثاني فهو وضع كلمة WORD خارج القوسين كالآتي :

PRINT WORD ("BIG "GER "WORD)



● توصيل الكلمات والقوائم في جملة SENTENCE

بهذا الأمر الجديد SENTENCE يمكن توصيل الكلمات والقوائم لتكوين قائمة كبيرة . ويستخدم هذا الأمر مُدخلين فقط وقد يكون أى منهما كلمة أو قائمة .

ولنر معاً هذه الأمثلة :

PRINT SENTENCE "A [WORD PLUS A LIST]

PRINT SENTENCE [A LIST PLUS A] "WORD

PRINT SENTENCE "TWO "WORDS

والأمر SENTENCE يتم اختصاره إلى SE كما في المثال الآتي لتوصيل قائمتين معاً :

PRINT SE [TWO LISTS] [MAKE A LIST, TOO]

● توصيل العديد من القوائم والكلمات (SENTENCE..)

وكما مع الأمر WORD فإن الأمر SENTENCE يمكنه أيضاً استخدام أكثر من مُدخلين (INPUT) إذا استخدمنا معه الأقواس العادية ().

● مثال :

PRINT (SENTENCE "THIS [WILL BECOME] [ONE LIST] "TOO)

● طباعة جزء من كلمة أو قائمة

FIRST, LAST, BUTFIRST, BUTLAST

هاك أربعة أوامر جُدد للتعامل مع الكلمات والقوائم وتقطيعها إلى أجزاء .

● الأمر الأول هو FIRST وهو يستخدم لطباعة الحرف الأول من الكلمة أو الكلمة الأولى من القائمة .

● الأمر الثاني هو LSDY ويستخدم لطباعة آخر حرف في كلمة أو آخر كلمة في قائمة .

● الأمر الثالث هو BUTFIRST بمعنى « ما عدا الأول » أى طبع كل الكلمة ما عدا الحرف الأول . أو طبع كل القائمة ما عدا الكلمة الأولى .

● الأمر الرابع هو BUTLAST بمعنى « ما عدا الأخير » أى طبع الكلمة ما عدا الحرف الأخير . أو طبع القائمة ما عدا الكلمة الأخيرة .

فلنر الأمثلة مع الكلمة :

PRINT FIRST "HELLO
PRINT LAST "HELLO
PRINT BUTFIRST "HELLO
PRINT BUTLAST "HELLO

● مثال ١ :

نتيجة تنفيذ المثال هي : طبع الحرف H نتيجة للأمر الأول . يليه طبع الحرف O نتيجة للأمر الثاني على سطر جديد : يليه طبع الجزء ELLO من كلمة HELLO . ثم الجزء HELL على السطر الرابع .

أى أن نتيجة التغير كالآتى :

H ←	نتيجة الأمر الأول	« أول حرف .. »
O ←	نتيجة الأمر الثاني	« آخر حرف .. »
HLLO ←	نتيجة الأمر الثالث	« ما عدا الأول .. »
HELL ←	نتيجة الأمر الرابع	« ما عدا الأخير .. »

● مثال ٢ :

هنا نستخدم نفس الأوامر ولكن مع القوائم LISTS :

```
PRINT FIRST [HELLO MY FRIEND]
PRINT LAST [HELLO MY FRIEND]
PRINT BUTFIRST [HELLO MY FRIEND]
PRINT BUTLAST [HELLO MY FRIEND]
```

وتكون نتيجة التنفيذ هي :

HELLO ←	أول كلمة ..
FRIEND ←	آخر كلمة ..
MY FRIEND ←	ما عدا أول كلمة ..
HELLO MY ←	ما عدا آخر كلمة ..

● مثال ٣ :

ومن الممكن إجراء تركيبات مختلفة من هذه الأوامر « لتقطيع » الكلمة أو القائمة لانتقاء جزء معين منها كالآتى :

PRINT FIRST BUTFIRST [HELLO MY FRIEND]
PRINT BUTFIRST FIRST [HELLO MY FRIEND]

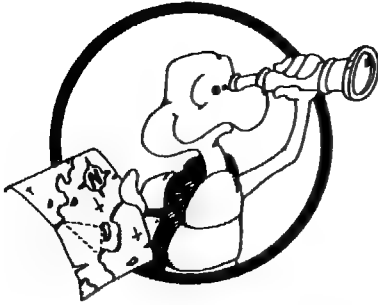
● الأمر الأول ينتج عنه طبع كلمة MY لماذا ؟

أن الأمر BUTFIRST سيقوم بطبع القائمة MY FRIEND تاركاً الكلمة الأولى . ثم يأتي الأمر FIRST ليطلع الكلمة الأولى وهي MY .

● أما الأمر الثاني فينتج عنه طبع الكلمة ELLO كالآتي :

الجزء الأول من الأمر وهو FIRST سيطبع كلمة HELLO كاملة وعندما يأتي دور الأمر BUTFIRST سيجد أمامه كلمة واحدة فيطبع منها كل الحروف ما عدا الأول .

تجربة



● هل تستطيع الوصول إلى تركيبة معينة من الأوامر السابقة لتجزئة القوائم والكلمات بحيث تطبع دائماً العنصر الثالث من القائمة أو الكلمة ؟

● ماذا أيضاً عن العنصر الرابع والخامس .. إلخ ؟

● جرب أيضاً طبع العنصر الأول والثاني والثالث من المؤخرة ..

ومن الجدير بالذكر أن بعض الأوامر الأخيرة أيضاً يمكن اختصارها كالآتي :

الاختصار	الأمر
BF BL	BUTFIRST BUTLAST

●● أمثلة عامة ●●

READLIST

● الكمبيوتر يسأل وأنت تجيب

● مثال [١] :

اكتب هذا البرنامج الفرعى وشاهد النتيجة :

الكمبيوتر ينتظر منك ..
TO TALK
PRINT [PLEASE TYPE SOMETHING FOR ME TO SAY]
PRINT SENTENCE [YOU JUST MADE ME SAY] READLIST
END

عند تشغيل هذا البرنامج الفرعى سوف يطبع الكمبيوتر على الشاشة الرسالة التى فى القائمة الأولى وهى :

PLEASE TYPE SOMETHING FOR ME TO SAY

بمعنى اكتب لى شيئاً لأقوله ..

وسوف ينتظر منك كتابة أى شىء على الشاشة ..

فإذا ما كتبت شيئاً وليكن كلمة "hello" ثم أدخلتها بالضغط على الزر "ENTER" فإنه يستقبل هذه الكلمة وعلى الفور يطبع الرسالة الآتية :

YOU JUST MADE ME SAY hello

بمعنى : لقد جعلتنى أقول hello .

والأمر الجديد READLIST هو الذى يجعل الكمبيوتر يتوقف لاستقبال قائمة ما وتخزينها فى القائمة READLIST .

ولذلك فعند تنفيذ الأمر الثانى فى البرنامج الفرعى فإنه يطبع القائمة

الأولى يليها القائمة المخزنة في READLIST .
● ماذا يحدث بداخل الكمبيوتر ؟

الأشكال التالية من (أ) إلى (ز) تشرح بالتفصيل ما يحدث من عمليات بداخل الكمبيوتر عند تنفيذ السطر الثاني من البرنامج :

PRINT SENTENCE [YOU JUST MADE ME SAY] READLIST

هذا السطر يتكون من ثلاثة أوامر لوجو وهي :

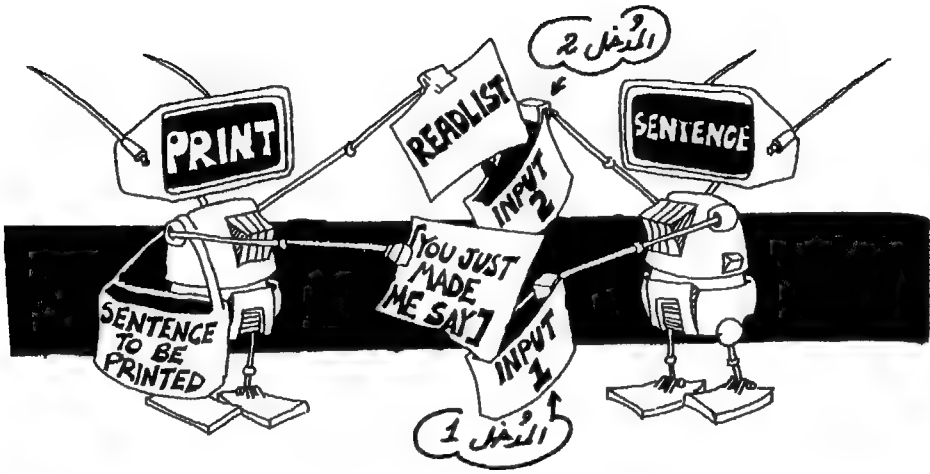
. READLIST ، SENTENCE ، PRINT

وكل أمر من هذه الأوامر يحتاج لبعض المُدخلات (inputs) لكي يؤدي العمل المطلوب منه . فالأمر PRINT مثلاً يحتاج إلى مُدخل يخبره بالشيء المطلوب طباعته على الشاشة .

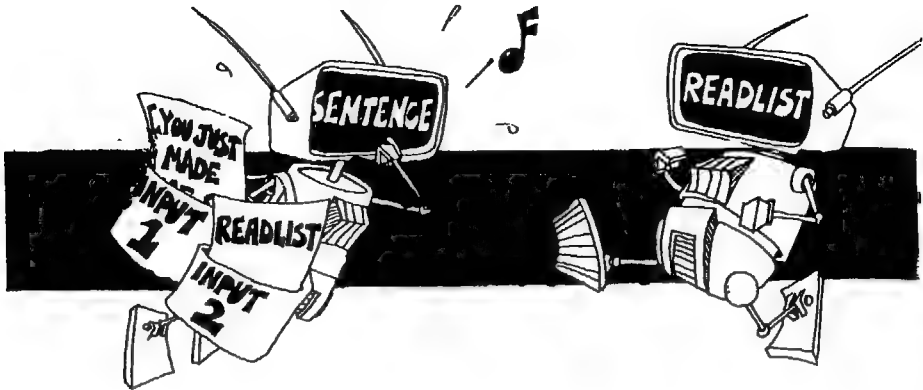
وفي الشكل تم تمثيل كل أمر من هذه الأوامر بروبوط يسعى لجمع المُدخلات اللازمة له حتى يستطيع أن يؤدي عمله .. فلنر معاً ماذا يحدث لتنفيذ السطر الثاني من البرنامج :



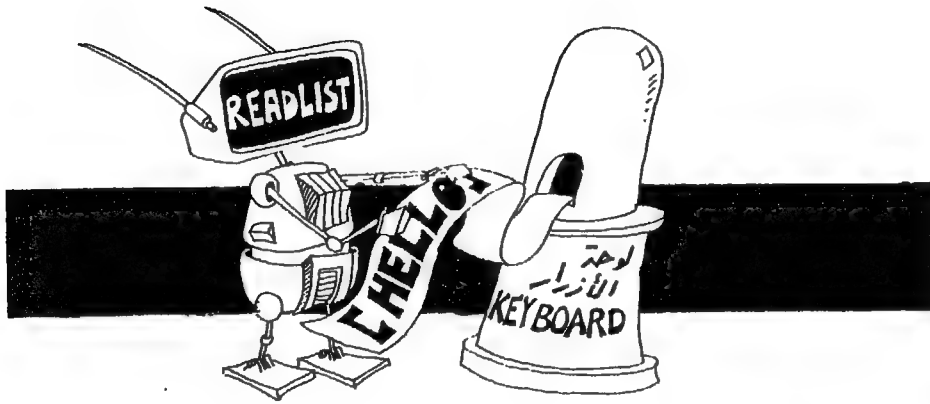
(أ) يحتاج الأمر PRINT لمُدخل ما (input) لكي يعرف الكمبيوتر ماذا سيُطبع . وهو يحصل على هذا المُدخل من الأمر SENTENCE . وهذا هو سر الترحيب الذي يبدو على الروبوط PRINT عندما عثر على المُدخل . SENTENCE



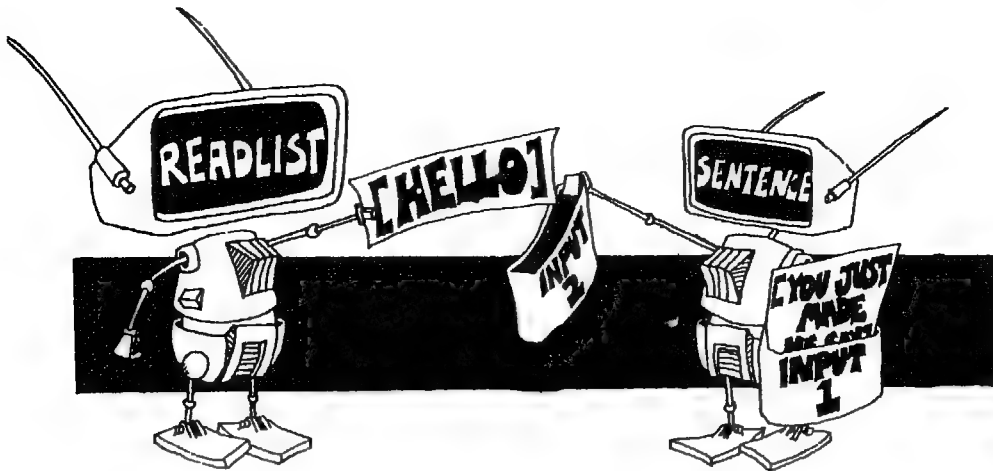
(ب) ولكن الأمر SENTENCE يحتاج هو الآخر لمُدخلين . وفي مثالنا هذا يكون المُدخل الأول هو القائمة [YOU JUST MADE ME SAY] أما المُدخل الثاني فهو الأمر READLIST . القائمة جاهزة ولكن الأمر READLIST يجب البحث عنه .



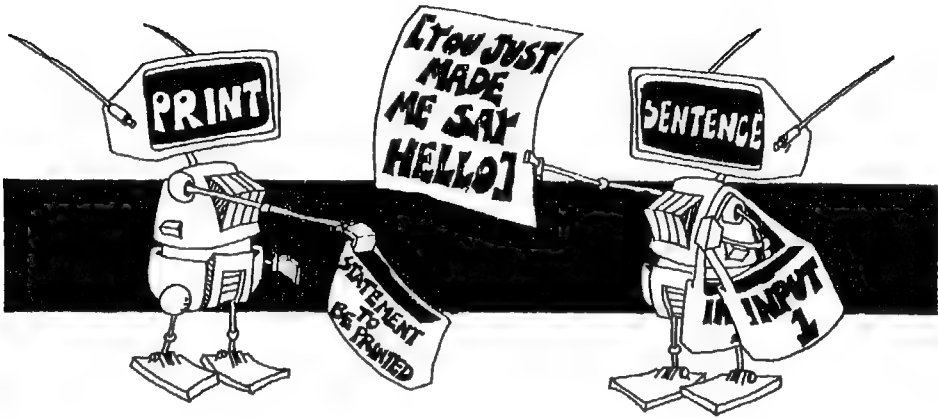
(ج) ها هو الروبوت SENTENCE يعثر على ضالته المنشودة . READLIST



(د) ولكن REALIST يحتاج هو الآخر لمُدخل من لوحة الأزرار حتى يصلح لأداء وظيفته .. وها هو يحصل على المدخل HELLO من لوحة الأزرار .



(هـ) الروبوت READLIST يسلم SENTENCE القائمة [HELLO] ليضعها في حقبة المُدخل رقم 2 .



(و) اكملت الآن القائمة التي يحتوى عليها SENTENCE لذلك فهو
يسلمها إلى PRINT لطبعها على الشاشة .



المرحلة الأخيرة من القصة .. نعرفها نحن جيداً .. بضغطه واحدة على
الزر ENTER تطبع العبارة المطلوبة على الشاشة ... هكذا ببساطة .. ربما
دون أن نحس بقيمة المجهود الذى بذله كل من READLIST
و SENTENCE و PRINT !

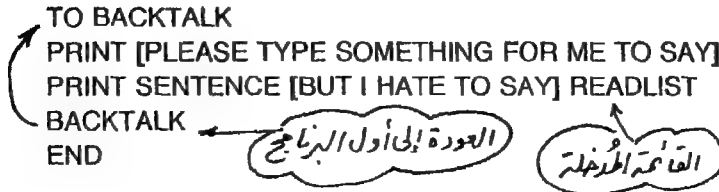
● الكمبيوتر يناقش !

● مثال ٢ :

وهنا نضيف إضافة صغيرة تجعل الكمبيوتر — بعد أن ينفذ العمل المطلوب منه — يعود إليك من جديد قائلاً :

« ولكنني أكره أن أقول هذه الكلمة أعطني شيئاً آخر لأقوله » .

وكلما أدخلت إليه عبارة عاد من جديد .. يعترض ويريد عبارة أخرى . وهذا هو البرنامج الفرعى وهو يحمل الاسم BACKTALK :



كيف يعود الكمبيوتر إلى أول البرنامج ليسألك مرة بعد مرة أن تدجّل له جملة يقولها ؟

إن هذا قد تم بفضل الأمر الذى جاء فى السطر قبل الأخير وهو اسم البرنامج الفرعى نفسه BACKTALK بمعنى تنفيذ البرنامج من البداية .

وهذه الحلقة التكرارية (loop) لا تتوقف إلا إذا تدخلت فى البرنامج من الخارج بالاستعانة بالزر Ctrl-Break وذلك للكمبيوتر IBM أو الزر المناسب فى أى كومبيوتر آخر .

● الكمبيوتر موافق دائماً :

● مثال ٢ :

فى هذا المثال فالكمبيوتر يوافق على ما تقدم له فهو يطلب منك أن تكتب له شيئاً تحبه وشيئاً تكرهه وفى الحالتين يبدو موافقاً تماماً على رأيك لكنه أبداً

يعود من جديد ليسألك أن تكتب له شيئاً آخر .

في هذا البرنامج تدريب جيد لاستخدام الأمر SENTENCE لوصل القوائم
معاً بما في ذلك القائمة التي يحتوى عليها أمر الإدخال READLIST .

```
TO AGREE
PRINT [TELL ME SOMETHING YOU LIKE]
PRINT (SENTENCE [I LIKE] READLIST [TOO])
PRINT [TELL ME SOMETHING YOU HATE]
PRINT (SENTENCE [I HATE] READLIST [EVEN MORE THAN YOU DO!])
AGREE
END
```

وهذا هو تنفيذ البرنامج على الكمبيوتر سنكلير :

تسغيل البرنامج الفرعى

الكمبيوتر يسأل

?AGREE ← tell me something you like

أنت تكتب الإجابة

APPLES ←

تعليق الكمبيوتر ← I like APPLES too

السؤال الثاني

please type something you hate

أنت تجيب ← TO DO NOTHING

تعليق الكمبيوتر ← I hate TO DO NOTHING even more than you do !

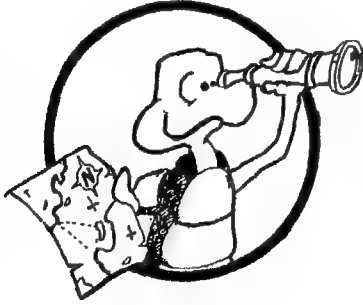
تسأل من جديد ...

تسأل من جديد ...

STOPPED!!! in AGREE

أيقاف البرنامج من الخارج ؟

تجربة



الروبوط المعترض دائماً

حاول إنشاء برامج مماثلة للحوار
بينك وبين الكمبيوتر باستخدام
نفس المنطق . مارأيك برنامج يجعل
الكمبيوتر دائم الاعتراض على كل
شيء ؟

إنه نموذج جيد لروبوط
مشاكس .

٦

استخدام المتغيرات
(variables)

MAKE

● ما معنى المتغير ؟

المتغير هو وعاء في الذاكرة نطلق عليها اسماً ما ونستخدمه لتخزين ما نشاء من البيانات سواء كانت هذه البيانات أرقاماً أو كلمات أو قوائم .

واسم المتغير في لغة لوجو عبارة عن كلمة (WORD) .

ولتر معاً هذا المثال :

● مثال :

إذا أردنا أن نخزن الرقم 8 في خانة في ذاكرة الكمبيوتر فعلينا أولاً أن نمنح هذه الخانة اسماً ، وليكن AGE . ثم نعطي الأمر الآتي :

? MAKE "AGE 8

مسافة خالية

بهذا الأمر تكون الكلمة AGE محتوية على العدد 8 .

وللتأكد من ذلك فإننا يمكن أن نطلب من الكمبيوتر أن يطبع محتويات خانة الذاكرة المسماة AGE (أو محتويات الكلمة AGE) بالأمر الآتي :

? PRINT :AGE

مسافة خالية

لاحظ هذه العلامة

سوف يطبع الكمبيوتر الإجابة فوراً تحت الأمر مباشرة كالاتي :

? PRINT :AGE

8

الإجابة

● اختر الاسم المناسب للمتغير :

ولكن لماذا اخترنا الاسم AGE اسماً للمتغير ؟ في الواقع أنه يمكن اختيار أي اسم . ولكن إذا كان الرقم 8 يعبر عن عمر أحد الأصدقاء فيكون من المناسب اختيار الاسم AGE للمتغير الذي نحفظ فيه الرقم حتى يكون معبراً عما يحتويه . تماماً كما نحفظ السكر في علبة مكتوب عليها سكر . والملح في علبة مكتوب عليها ملح . ولا ضرر إطلاقاً إذا كتبنا على علبة السكر « ملح »

وكتبنا علي علبة الملح « سكر » . ولكن ربما إذا لم نتذكر ذلك جيداً أن نضع
الملح على الشاي حسب اسم المتغير المكتوب على العلبة !

● الكلمات والقوائم في متغيرات أيضاً :

اكتب هذا البرنامج على الشاشة :

```
TO VARS
MAKE "AGE 8
MAKE "NAME [ SALLY OSSAMA ]
MAKE "TEL "5865372
```

في هذا البرنامج نضع السن (8) في المتغير AGE ونضع قائمة الاسم
[SALLY OSSAMA] في المتغير NAME أما رقم التليفون (5865372) فنحزنه
ككلمة في المتغير TEL .

فالسن تم تخزينه كعدد .

والاسم تم تخزينه كقائمة لذلك وضعناه بين قوسين مربعين .

والتليفون تم تخزينه ككلمة لذلك استخدمنا قبله العلامة " .

أما المتغيرات جميعاً فهي عبارة عن كلمات لذلك اسبقناها كلها بالعلامة «

(ومن الممكن أيضاً تخزين العدد 8 ككلمة باستخدام العلامة " قبله). بتنفيذ
هذا البرنامج بإعطاء الأمر VARS لا يتغير شيء على الشاشة ولكن أشياء حاسمة
تأخذ مجراها بداخل الكمبيوتر حيث يتم تخصيص الرقم 8 للمتغير AGE ويتم
تخصيص قائمة الاسم SALLY OSSAMA للمتغير NAME ويتم تخصيص
الكلمة 5865372 للمتغير TEL .

ويمكن التأكد من ذلك بإعطاء الأوامر الآتية بالطريقة المباشرة :

الأمر ← ? PR :NAME

الإجابة ← SALLY OSSAMA

الأمر ← ? PR :AGE

الإجابة ← 8

الأمر ← ? PR :TEL

الإجابة ← 5865372

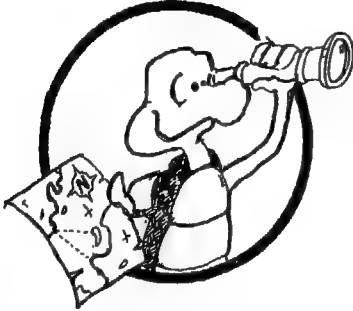
والعدد المخزن في الذاكرة كمتغير يمكن إجراء عليه ما نشاء من العمليات الحسابية . فمثلاً يمكن أن نكتب الأمر الآتي :

? PR :AGE + 7

الإجابة ← 15

في هذه الحالة يستبدل الكمبيوتر المتغير AGE بالعدد 8 المخزن فيه ويجمعه على العدد 7 .

تجربة



هل حاولت جمع عدد ما على رقم التليفون أيضاً (وهو مخزن ككلمة) .. هل يتحقق ذلك أم يشكو الكمبيوتر ؟

جرب .. وتحقق من النتائج .

في حديثنا السابق استخدمنا كلمة التخصيص بدلاً من التخزين وهذا هو الاسم الرسمي لعملية تخزين بيانفي متغير .

هل يمكن أن نتحدث الآن عن تخصيص متغير لمتغير ؟ بمعنى تخزين قيمة متغير ما لمتغير آخر باسم مختلف ؟

● تخصيص متغير لمتغير :

يمكن إجراء ذلك بأمر كالآتي :

? MAKE "TELEPHONE :TEL

فإذا كان الكمبيوتر يعرف قيمة المتغير TEL فإنه يخصصها للمتغير الجديد TELEPHONE ويمكن التأكد من ذلك بطباعة قيمة المتغير الجديد TELEPHONE كالآتي :

? PR :TELEPHONE

فلاش

هل هناك ما يحيرك مع استخدام العلامتين (") و (:) ؟
إننا عند التخصيص بالأمر MAKE نستخدم العلامة الأولى .
وعند الطباعة نستخدم الثانية .
ويمكن لعدم الخلط بينهما : فهم العلامة الأولى على أنها « اسم » للمتغير . أما العلامة الثانية فهي تعني « قيمة » المتغير .
وبذلك يمكن قراءة الأمر الأخير على أنه : « خصص قيمة المتغير TEL لاسم المتغير TELEPHONE » .



ومن الممكن استبدال العلامة (:) أمام اسم المتغير بكلمة (THING) وهى
نستخدم كالاتى :

الأمر ← ? PRINT THING *AGE
الإجابة ← 8

● طباعة العناوين :

قد نرغب فى طبع بعض العناوين أمام البيانات بالصورة الآتية على سبيل
المثال :

NAME: SALLY OSSAMA
AGE: 8

TEL: 5865372



يمكن إجراء ذلك باستخدام خصائص القوائم التى عرضناها من قبل. فلطباعة
العنوان (NAME:) أمام بيان الاسم نكتب الأمر الآتى :

الأمر ← ? PR SE [NAME:] :NAME
العنوان ← قيمة المتغير أو البيان
الإجابة ← NAME: SALLY OSSAMA

وقد تصادف هنا أن يكون العنوان هو (NAME:) ولكن هذا العنوان لا
يتم بصلة للمتغير (NAME) وقد كان من الممكن استخدام أى عنوان آخر مثل
(NAME OF STUDENT) .

وبالمثل يمكن طباعة بقية البيانات مثل السن والتليفون وأمام كل منها العنوان
المناسب .

تمرين



حاول طباعة جميع البيانات
والعناوين بأمر واحد باستخدام
الأمر PRINT SE

Functions

● الدوال

هنا نوعية من البرامج تسمى « برامج الدوال » أو « الدوال » . وفي هذه
النوعية من البرامج نستخدم المتغيرات ولكن عملية التخصيص للمتغير تتم
بطريقة مختلفة تماماً عن الأمر MAKE ولنبدأ بهذا المثال .

● مثال :

اكتب برنامج الدالة الآتي :

TO SQUARE :SIZE (المتغير)
REPEAT 4 [FORWARD :SIZE RIGHT 90]
END

هذا هو برنامج المربع (الصندوق) الذي رسمناه من قبل ولكن ظهرت به
أشياء جديدة نريد التوقف عندها ، فقد ظهرت كلمة SIZE مرتين بالبرنامج .
فلاحظ أن الأمر FORWARD لم يعقبه رقم ما كالعادة بل تبعته الكلمة
SIZE وهذه الكلمة هي اسم المتغير ، وهو يعبر هنا عن طول ضلع المربع .
وطول ضلع المربع هنا ليس عدداً ثابتاً بل من الممكن تحديده في كل مرة
يُنفذ فيها البرنامج . وهذا هو الغرض من برنامج الدالة .

ويتميز برنامج الدالة أيضاً بوجود المتغير ضمن الاسم :

TO SQUARE :SIZE

وعند تشغيل هذا البرنامج (أو هذه الدالة) فإن الكمبيوتر ينتظر منك إدخال قيمة المتغير SIZE وبمجرد إدخال هذه القيمة يستخدمها في رسم المربع .

أى أن تشغيل هذا البرنامج يتم بأمر كأحد الأمثلة الآتية :

```
SQUARE 30
SQUARE 40
SQUARE 50
```

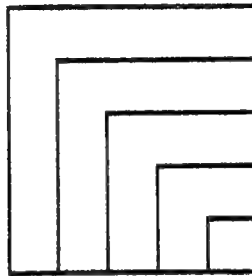
وعند إعطاء الأمر الأول يقوم الكمبيوتر برسم مربع طول ضلعه ٣٠ . أما مع الأمر الثانى فيرسم مربع طول ضلعه ٤٠ وهكذا .. ولكن باستخدام الأمر SQUARE بدون أى رقم بعده يؤدي إلى رسالة خطأ كالمثال الآتى :

? SQUARE

Not enough inputs to square

رسالة الخطأ

وبتشغيل هذا البرنامج مرة بعد باستخدام قيمة جديدة للمتغير SIZE يمكن الحصول على شكل كالتالى :



تنفيذ الدالة SQUARE مع تغيير قيمة الضلع كل مرة

والمتغير المستخدم مع الدالة يطلق عليه غالباً « الدليل » (argument) أو « دليل الدالة » .

● ارسم مثلثاً متغير الضلع :

هذا البرنامج يرسم مثلثاً متساوي الأضلاع ، وطول ضلعه يمثل المتغير SIZE الذى يتم إدخاله كل مرة يتم فيها تنفيذ البرنامج .

طول الضلع
TO TRIANGLE :SIZE
REPEAT 3 [FORWARD :SIZE RIGHT 120]
END

نفذ هذا البرنامج عدة مرات تحصل على مجموعة من المثلثات المتداخلة معاً .
مع ملاحظة إخفاء السطح قبل الرسم أو وضع الأمر HT بداخل البرنامج .



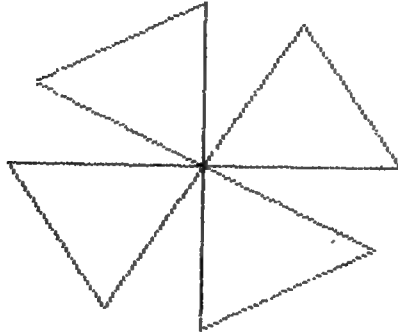
جرب الآن هذا التعديل :

TO TRIANGLE :IZE
RT 30 FD :SIZE
REPEAT 2 [RT 120 FD :SIZE]
END

في هذا البرنامج قد أضفنا تعديلاً حتى يُرسم المثلث الموضح بعد و يبدأ الرسم بتغيير الاتجاه إلى اليمين بمقدار 30 درجة .



جرب الآن تنفيذ هذا البرنامج أربع مرات متتالية بنفس طول الضلع SIZE
تحصل على الشكل التالى .



تنفيذ البرنامج ٤ مرات بطول ضلع ٧٠
منفذ على كومبيوتر سنكلير

Recursion

● نداء الدالة لنفسها

تتميز لغة لوجو بخاصية فريدة وهى إمكانية تشغيل البرنامج من داخله .. فالبرنامج السابق للدالة التى ترسم مثلثاً يمكن تشغيله عدة مرات لنحصل على المثلثات المتعددة .

كما يمكن إعطاء الأمر الآتى مرة واحدة لتكرار التنفيذ :

REPEAT 4 [TRIANGLE 70]

فيرسم المثلث ٤ مرات .

أما الشيء الجديد الذى يمكن عمله فهو إضافة سطر أخير فى البرنامج عبارة عن اسم برنامج الدالة نفسه كالاتى :

كوميبيوتر سنظير

```
?
TO TRIANGLE :SIZE
AT 30 FD :SIZE
REPEAT 2 [RT 120 FD :SIZE]
TRIANGLE :SIZE
END
```

نداء الدالة

بهذا يتم تشغيل برنامج الدالة بصفة دائمة . فالبرنامج يبدأ برسم المثلث مرة واحدة ثم يأتي للسطر الأخير الذي يرسله إلى أول الدالة فينفذ أوامر البرنامج من البداية وهكذا ..

ولن يتوقف هذا البرنامج إلا بالتدخل من الخارج بالضغط على زر التحكم CTRL + break (في الكوميبيوتر IBM) أو على أحد الأزرار المكافئة في الأجهزة الأخرى .

ولو أن السلحفاه كانت ظاهرة أثناء تنفيذ البرنامج لوجدت أنها تدور في حلقة مقفلة وهي ترسم نفس الشكل مرّة بعد مرّة .
ولعلنا نلاحظ أنه لا يمكن رسم أكثر من أربعة أشكال في الشاشة الواحدة أما مازاد عن ذلك فإنه يكون مجرد إعادة للرسم وذلك لأن كل مثلث يحتل ٦٠ درجة علاوة على ٣٠ درجة فاصلة بينه وبين المثلث المجاور وبذلك تحتل المثلثات الأربعة $4 \times 90 = 360$ درجة .

● مازلنا نلعب بالمثلثات المتغيرة :

● مثال :

نقدم هنا تطبيقاً آخر للدوال المتكررة التي تستدعى نفسها ولكننا هذه المرّة سوف نحصل على شكل مختلف عند كل تنفيذ للبرنامج .

والبرنامج يتكون من ثلاثة برامج في الحقيقة :
البرنامج الأول لإعداد الشاشة ويسمى SETUP :

TO SETUP

CS ← تنظيف الشاشة

LT 30 ← الاستدارة إلى اليسار ٣٠ درجة

HT ← إخفاء السلخفاء

END

البرنامج الثاني برنامج فرعي لرسم مثلث ويسمى TRI (اختصار كلمة
TRIANGLE) :

TO TRI

REPEAT 3 [FD 80 RT 120]

END لرسم مثلث طول ضلعه ٨٠

البرنامج الثالث برنامج دالة يستدعى البرنامج السابق TRI كما يستدعى نفسه
ويسمى PAT (اختصار كلمة PATTERN) :

TO PAT :N

TRI ← استدعاء البرنامج الثاني

RT 360 / :N ← الدوران لليمين

PAT : N ← استدعاء الدالة لنفسها .. للتكرار

END

أما البرنامج الأول فهو لمجرد إعداد الشاشة وإدارة السلحفاه بمقدار ٣٠ درجة جهة اليسار وعند تنفيذه بالأمر SETUP لن نشاهد شيئاً على الشاشة لا سيما أن السلحفاه تكون مخفية أيضاً .

مند تشغيل البرنامج الثانى TRI سوف يرسم مثلثاً متساوى الأضلاع قاعدته لأعلى ورأسه لأسفل عند بيت السلحفاه . وطول ضلعه ٨٠ خطوة كما هو موضح بالبرنامج وهذا بعد تنفيذ البرنامج (SETUP) .



أما البرنامج الثالث فهو برنامج للدالة يستدعى البرنامج TRI ويستخدمه ضمن خطواته ولذلك يجب تشغيل هذا البرنامج بعد البرنامج SETUP مباشرة . والدالة PAT تستدعى البرنامج TRI لرسم المثلث السابق أولاً ثم تدير السلحفاه بزاوية مقدارها $(360/N)$ حيث N هو المتغير الذى نكتب قيمته عند تشغيل البرنامج PAT كالمثال الآتى :

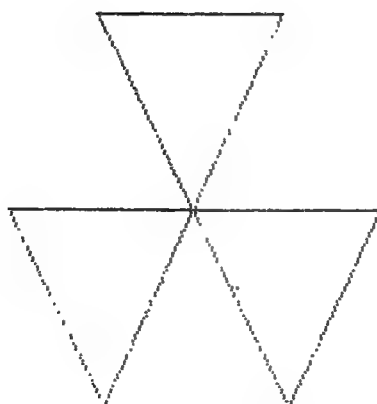
PAT 1

PAT 6 أو

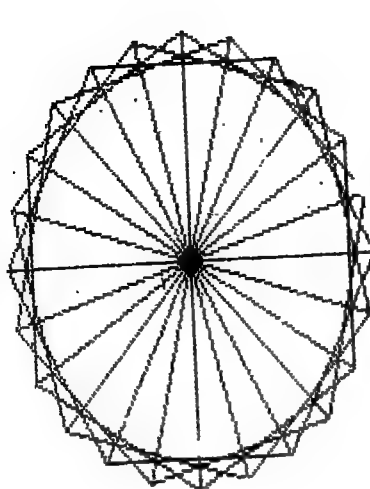
PAT 8 أو

معنى الدوران بزاوية $(360/N)$ أنه يتم تقسيم الزاوية الدائرية كلها إلى عدد N من الأقسام . فإذا أعطينا الأمر الأول PAT 1 سوف نحصل على مثلث واحد مماثل للمثلث الذى نحصل عليه من البرنامج TRI . وإذا أعطينا الأمر PAT 2 نحصل على مثلثين . والأمر PAT 3 يرسم ثلاثة مثلثات وهكذا ..

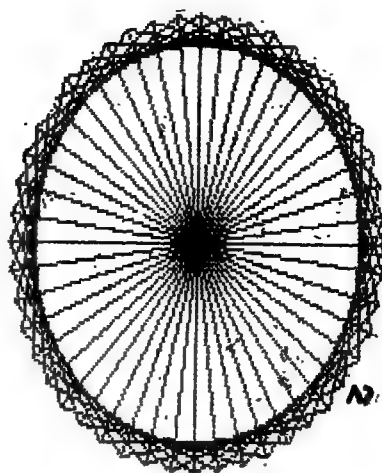
. جرب الآن تشغيل البرنامج بالأوامر الموضحة وحاول الحصول على الأشكال التالية بعد بتغيير الرقم N .



PAT 3



PAT 24



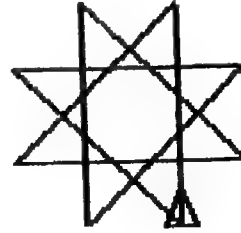
PAT 48

● التكرار مع البرامج أيضاً :

وخاصية التكرار ممكنه مع البرامج العادية أيضاً كما أشرنا من قبل . والمثال الآتي يرسم النجمة ذات الثمانية أضلاع بمجرد رسم أول ضلع لها ثم تحديد زاوية الدوران وإطلاق البرنامج ليكرر نفسه بلا نهاية . والبرنامج اسمه STAR135 باعتبار أن زاوية الدوران لهذه النجمة ١٣٥ درجة .

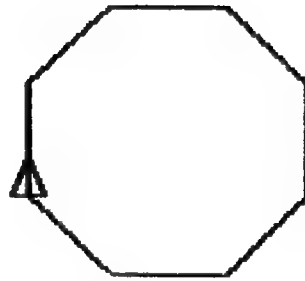
TO STAR135
FORWARD 60
LEFT 135
STAR135
END

← تكرار البرنامج



وهذا البرنامج يرسم شكل ثمانى الأضلاع وزاوية الدوران المستخدمة هنا هي (360/8) أى ٤٥ درجة . وهو يرسم أول ضلع ويدير السلحفاة بالزاوية المطلوبة ثم يكرر نفسه .

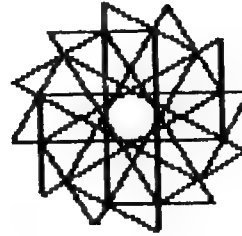
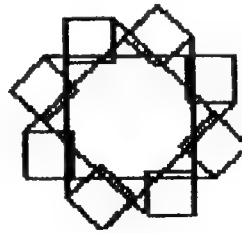
TO STAR45
FORWARD 30
RIGHT 45
STAR45
END



تمرين



يمكنك استغلال خاصية التكرار
لرسم الأشكال الآتية بعد من
الوحدات الأساسية الموضحة على
اليسار .

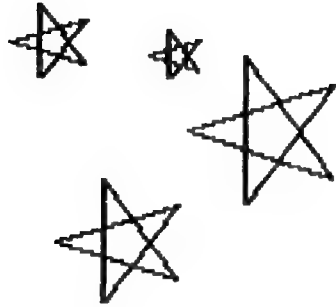


● ارسم نجمة خماسية بأحجام مختلفة :

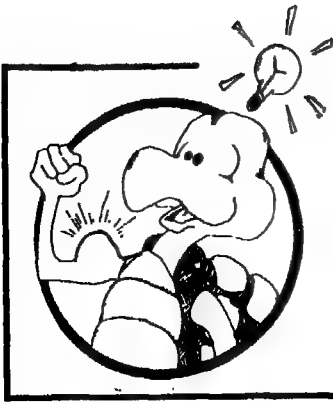
اكتب البرنامج التالي لرسم النجمة الخماسية :

```
TO STAR5 :SIZE
REPEAT 5 [FORWARD :SIZE RIGHT 144]
END
```

ويعمل هذا البرنامج عندما تُدخل الأمر STAR5 متبوعاً برقم ما مثل 50 أو 60 ... إلخ ، يمثل طول ضلع النجمة الخماسية . عندئذ نحصل على أحد الأشكال المرسومة بعد .



فكرة



هل تستطيع رسم هذه المجموعة من
النجوم على شاشة واحدة ؟
كل ما يتطلبه الأمر هو مراجعة ما
درسناه معاً بدقة كافية !

● ماذا يحدث بداخل الكمبيوتر :

تحدثنا عن المتغيرات واستفصنا في تطبيقاتها سواء باستخدام أمر التخصيص
MAKE أو باستخدام برامج الدوال التي ندخل إليها قيمة المتغير (الدليل) .

والأشكال الآتية بعد توضيح لنا ما يقوم به الساحر لوجو من أعمال عندما
نبدأ في تعليم الكمبيوتر بالأمر SIZE: TO SQUARE وحتى يتم تنفيذ
البرنامج .



(أ) بعد الانتهاء من كتابة برنامج الدالة (SQUARE : SIZE) فإن الساحر لوجو يحتفظ بخطوات البرنامج في الذاكرة . ويخصّصُ خانة خالية للمتغير "SIZE" ولنطق عليها خانة الدليل .



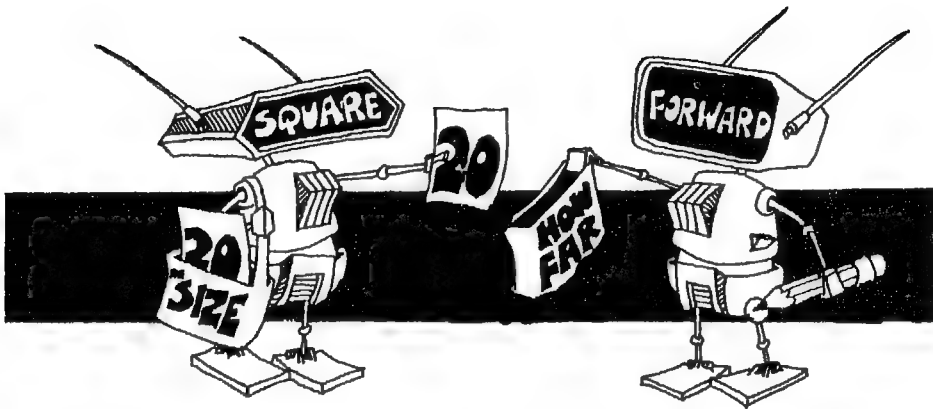
(ب) عندما نعطى أمر تشغيل برنامج الدالة مثل :

SQUARE 20

فإن الساحر لوجو يقوم بوضع الرقم 20 في خانة الدليل ثم ..




(ج) يقوم الساحر لوجو باستدعاء الروبوت SQUARE ويسلمه نسخة من قيمة المتغير "SIZE" (أى الرقم 20) حتى يتمكن من تنفيذ العمل المكلف به .



(د) أثناء تنفيذ برنامج الدالة SQUARE قد يصادف الروبوت SQUARE أمراً
مثل SIZE : FORWARD . عندئذ يستدعى الروبوت FORWARD
ويسلمه نسخة من قيمة المتغير SIZE ليستخدمها في أداء عمله .

فلاش



● تذكر الآتي :

"SIZE

يعنى اسم المتغير SIZE

:SIZE أما

فيعنى قيمة المتغير SIZE

SIZE أما

فهو الأمر الذى ندخله للكمبيوتر لكي ينفذ البرنامج SIZE إن وجد .

● مصطلحات : العلامة (") تسمى بالإنجليزية quotes وبالتالي فإن التعبير "Z يمكن قراءته : quotes Z .

أما العلامة (:) فهي تسمى dots وبالتالي فإن تعبيراً مثل N : يمكن قراءته : dots N .

● الفرق بين المتغير العام ومتغير الدالة :

رأينا في مستهل الموضوع عن المتغيرات أنه بعد تنفيذ البرنامج المحتوى على الأوامر MAKE ، يمكن طبع قيمة المتغيرات التي تم التخصيص لها بيانات مختلفة . فمثلاً قد أعطينا الأمر :

? PRINT :NAME

SALLY OSSAMA

?

وكان الجواب حاضراً .

وليس هذا هو الحال مع المتغيرات التي تستخدمها الدالة كدليل لها . بمعنى أنه لا يمكن بعد تنفيذ البرنامج **SIZE** : **SQUARE** أن نطبع قيمة المتغير **SIZE** . فهو يفقد القيمة المخصصة له بمجرد الانتهاء من التنفيذ .

ولنجرب الآتي :

نفذ البرنامج بالأمر :

SQUARE 50

ثم أعط الأمر التالي :

PRINT :SIZE

سوف تكون الإجابة هي رسالة خطأ مثل :

SIZE has no value

?

فما هو الفرق ؟

إن المتغير الذي خصصنا قيمته بالأمر **MAKE** هو متغير عام يحتفظ به الكمبيوتر في خانة عامة بحيث أنه يكون موجوداً دائماً في حيز العمل الذي نحتفظ فيه بكل البرامج والمتغيرات . أما المتغير المستخدم كدليل للدالة مثل **SIZE** فهو موجود في خانة تابعة للبرنامج كما رأينا مع الساحر لوجو . ولا يمكن الوصول لمحتويات الخانة **SIZE** إلا من خلال البرنامج أما بعد التنفيذ فإن محتويات الخانة **SIZE** تُفقد !

وهذه تجربة مثيرة يمكن بها التأكد من ذلك

أعط الأمر التالي :

MAKE "SIZE 50

بهذا الأمر نكون قد كَوَّننا متغيراً عاماً بالاسم SIZE ووضعنا فيه القيمة 50 . وهو يحمل نفس اسم الدليل SIZE .

وللتأكد من وصول القيمة 50 إلى خانة المتغير العام SIZE أعط الأمر التالي :

? PR :SIZE

الأمر

50

الإجابة

?

في نفس الوقت يمكننا تشغيل برنامج الدالة SQUARE الذى يستخدم الدليل SIZE كالتالى :

SQUARE 20

بهذا الأمر يتم تنفيذ البرنامج مع تخصيص القيمة 20 للمتغير الدليل SIZE .

ماذا تعتقد لو أننا أعطينا الأمر (SIZE : PR) الآن ؟

? PR :SIZE

50

?

إن المتغير العام SIZE ما زال يحتفظ بقيمته 50 ولم يتأثر بالمتغير الدليل SIZE الذى خصصنا له القيمة 20 أثناء تشغيل البرنامج . هذا هو الفرق .

● الدالة بأكثر من دليل :

يمكن للدالة أن تأخذ أكثر من دليل وفي هذه الحالة يقوم البرنامج الواحد برسم أكثر من شكل سواء من ناحية الحجم أو التصميم .

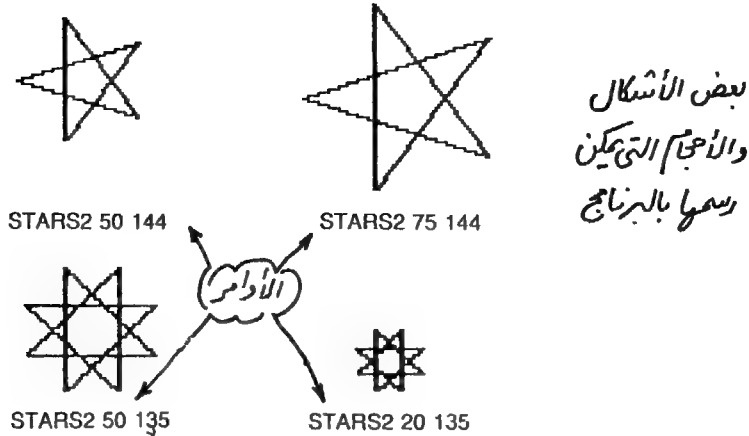
فمن الأمثلة التي ناقشناها قبلاً : رسم النجوم المختلفة فهناك النجمة ذات الأضلاع الخمسة التي تحتاج لزاوية دوران ١٤٤ درجة .

وهناك النجمة ذات الأضلاع الثمانية التي تستخدم معها الزاوية ١٣٥ والنجمة ذات الأضلاع التسعة بزاوية الدوران ١٥٠ درجة .

في الواقع أنه يمكن بناء برنامج واحد لرسم كل النجوم وباستخدام دليلين أحدهما يمثل متغير الزاوية والآخر يمثل طول ضلع النجمة — يمكننا رسم أى نجمة نرغبها :

```
TO STARS2 :SIZE :ANGLE
FORWARD :SIZE
RIGHT :ANGLE
STARS2 :SIZE :ANGLE
END
```

الزاوية ←
البحجم ←



وفي البرنامج الموضح يمثل الزاوية الدليل ANGLE ويمثل الحجم (طول الضلع) الدليل SIZE وعند تشغيل مثل هذا البرنامج يجب إدخال قيمة كل من المتغيرين كأمثلة الآتية :

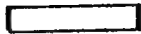
```
? STARS2 50 144
? STARS2 60 150
```

وموضح مع الأشكال التى تلى البرنامج الأوامر التى نتج عنها كل شكل منها .

أدخل ما تشاء من أرقام لهذا البرنامج لتشاهد كل أنواع النجوم الممكنة .
وهذا برنامج يمكن استخدامه لرسم أى شكل رباعى (مربع أو مستطيل)
بإدخال الأبعاد المطلوبة للشكل الطول (LENGTH) والعرض (WIDTH) .
وبلى البرنامج مجموعة من الأشكال التى يمكن الحصول عليها من البرنامج ومعها
الأوامر التى رسمتها .

```
TO RECTANGLE :LENGTH :WIDTH
FORWARD :LENGTH
RIGHT 90
FORWARD :WIDTH
RIGHT 90
FORWARD :LENGTH
RIGHT 90
FORWARD :WIDTH
RIGHT 90
END
```

العرض ←
الطول ←



RECTANGLE 10 50



RECTANGLE 60 20



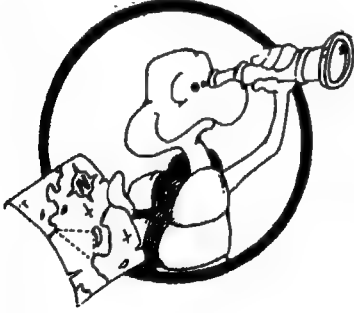
RECTANGLE 100 20

بعض الأشكال الناتجة من البرنامج



RECTANGLE 50 50

تجربة

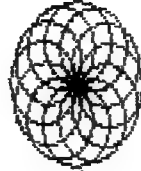


هل يمكنك تطوير البرنامج السابق
ليرسم أى شكل رباعى (معيّن أو
متوازى مستطيلات) علاوة على
المربع والمستطيل . حاول وسوف
تجد الحل فى الكتاب إذا عجزت .

تطبيقات

١ أشكال زخرفية بالدوائر :

قدمنا من قبل طريقة رسم هذا الشكل بالبرنامج العادى ، ونقدمه الآن
باستخدام برنامج الدالة التى تمنحه مرونة أكثر .



البرنامج الفرعى

```
?  
TO CM :S  
REPEAT 360 / :S [FD 1 LT :S]  
END
```

البرنامج الرئيسى

```
?  
TO CX  
REPEAT 12 [CM 4 RT 30]  
END
```

ولنبداً بالبرنامج الفرعى S : CM وهو يستقبل المتغير S الذى يمثل زاوية الدوران ويقوم برسم دائرة فى النصف الأيسر من الشاشة .
أما البرنامج الرئيسى فهو يستدعى البرنامج الفرعى ومعه الدليل فى نفس الوقت لذلك فهو يستدعيه بالأمر :

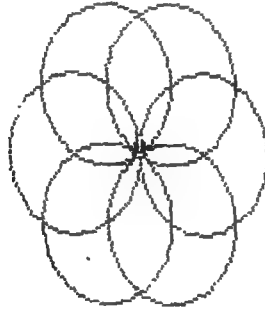
CM 4

أى أن الزاوية المستخدمة هى ٤ درجات .
وبعد رسم الدائرة تنحرف السلحفاة بمقدار ٣٠ درجة ويتكرر الرسم ١٢ مرة .

ولعلنا نلاحظ العلاقة بين الزاوية ٣٠ درجة وبين عدد مرات التكرار ، حيث أن الرحلة الكاملة ٣٦٠ لا تتسع لأكثر من ١٢ زاوية ٣٠ درجة .
ويمكن تكرار الرسم بطريقة أخرى وهى استدعاء البرنامج لنفسه بالأمر CX وذلك بدون تحديد لعدد مرات التكرار وفى هذه الحالة ستظل السلحفاة ترسم وتعيد ما رسمته إلى الأبد !

ولكن المنظر على الشاشة سيظل هو هو، ١٢ دائرة . وفى الحالة الأخيرة يوقف البرنامج بالضغط على الزر (Ctrl-Break) أو ما يعادله فى الأجهزة الأخرى (خلاف IBM) .

وإذا كانت الزاوية ٦٠ درجة بدلاً من ٣٠ درجة نحصل على الشكل الآتى الذى قدمناه من قبل وهو يتكرر ٦ مرات فقط .



٢ رسم دائرة ذات نصف قطر معين R :

يمكن لأصدقائنا هواة البحث والتقصي أن يرجعوا الآن إلى الملحق (١) لدراسة التفصيلات الرياضية لحساب نصف قطر الدائرة .
ومع ذلك فيمكن الآن أن نستخدم البرنامج الآتي لرسم دائرة ذات نصف قطر R .

```
TO C1 :R :A
MAKE "S 2*3.14159*:R/:A
REPEAT :A [ RT 180/:A FD :S RT 180/:A ]
END
```

حيث R : هو نصف قطر الدائرة المطلوب رسمها .
A : عدد الأضلاع التي تتكون منها الدائرة باعتبار أن الدائرة شكل مضلع . فمثلاً الدائرة التي رسمناها من قبل بالأمر :

```
REPEAT 360 [ FD 1 RT 1 ]
```

تتكون هذه الدائرة من ٣٦٠ ضلعاً طول كل ضلع خطوة واحدة . وأما الدائرة التي ترسم بالأمر :

```
REPEAT 360/10 [ FD 1 RT 10 ]
```

فهى تتكون من ٣٦ ضلعاً طول كل منهم خطوة واحدة أيضاً . وفي البرنامج RCIRCLE أو LCIRCLE للكمبيوتر IBM يستخدم دائماً ٣٦ ضلعاً لرسم أى دائرة . ولكن هذا البرنامج أكثر عمومية وبه حرية اختيار أكثر .

ومع ذلك فيمكنك. الإطلاع على البرامج المذكورة في الملحق رقم (١) واستخدامها كما يمكنك تخزينها على القرص المغنطيسى أو الشريط المغنطيسى واستدعائها عند الحاجة إليها .

وفي الصيغة التي قدمناها الآن نجد أن طول الضلع (S) مرتبط بعدد الأضلاع (A) ونصف القطر (R) وبالتالي فإن تغيير عدد الأضلاع (A) لا يغير نصف القطر وإنما يؤثر على درجة الدقة للرسم وعلى سرعة تنفيذه .
وهذا برنامج فرعى آخر يؤدي نفس الغرض ويعتمد على معالجة رياضية مختلفة :

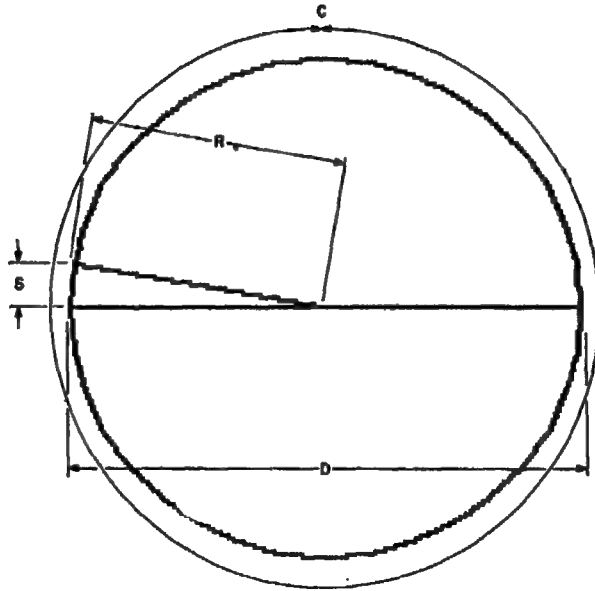
```
TO C2 :R :N
MAKE "S :R * SIN :N
REPEAT 360/:N [ RT :N/2 FD :S RT :N/2 ]
END
```

وفي هذا البرنامج تمثل R : قيمة نصف القطر المطلوب .
أما N : فهي الزاوية المواجهة لكل ضلع وهي طريقة مختلفة للتعبير عن عدد الأضلاع فإذا كانت هذه الزاوية ١٠ درجات مثلاً فإن الدائرة تحتوى على ٣٦ ضلعاً .

ويدل S: على طول الضلع .

وفي هذا البرنامج أيضاً لا تتأثر الزاوية (N) على مساحة الدائرة بل تؤثر على درجة الدقة والسرعة . ولنجرب إدخال القيمة (N) مساوية للرقم 40 مثلاً فسوف نجد أن الدائرة تحولت إلى شكل ثمانى .

ونلاحظ أن الزاوية يتم تقسيمها إلى نصفين مع الأمر RT وذلك لتحقيق الدقة المطلوبة (راجع الملحق (١)) .



الدائرة عبارة عن شكل مضلع

وللأصدقاء الذين لا يطبقون كثرة التعامل مع التفصيلات فإنه يمكن اختصار أى من البرنامجين السابقين بحيث لا ندخل له سوى نصف القطر فقط R فيصبح كالآتي :

برنامج فرعى لرسم دائرة قطرها R

```

2
TO C3 :R
MAKE "S 3.141592 * :R / 18
REPEAT 36 [RT 5 FD :S RT 5]
END

```

ويعمل هذا البرنامج بكتابة اسمه (C3) يليه نصف القطر المطلوب مثل :

? C3 40

هذا الأمر يرسم دائرة نصف قطرها 40 خطوة .

ولو أنك استخدمت البرنامج C1 فإن الأمر يصبح على سبيل المثال :

C1 40 36

هذا الأمر يرسم نفس الدائرة وبنفس الدقة .

أما البرنامج C2 فيمكن استخدامه كآلاتي :

C2 40 10

وهذا أيضاً يكافئ البرنامجين السابقين :



٣ رسم دائرة في منتصف الشاشة :

في البرامج السابقة تحدثنا عن الدائرة على اليمين والدائرة على اليسار والدائرة أعلى أو أسفل الشاشة . ولكننا لم نرسم بعد دائرة في منتصف الشاشة .

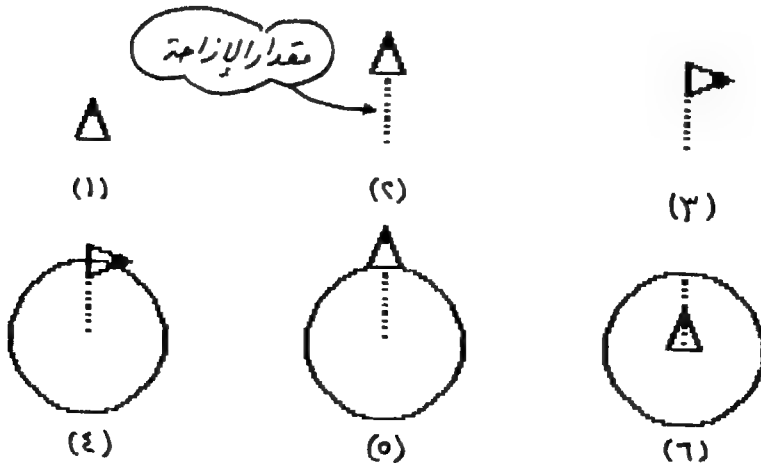
وحيث أن السلحفاه التي ترسم الدائرة تقع على محيطها باستمرار لذلك إذا أردنا أن تكون الدائرة في المنتصف تماماً فعلينا أن نحرّك السلحفاه قبل البدء في الرسم إلى المكان المناسب ! ولكن أين هو المكان المناسب ؟ كالعادة أماننا طريقان :

المحاولة والخطأ أو الحسابات

ولكن أياً كانت الطريقة المتبعة فهناك اتجاه معين لتحريك السلحفاه . فلو

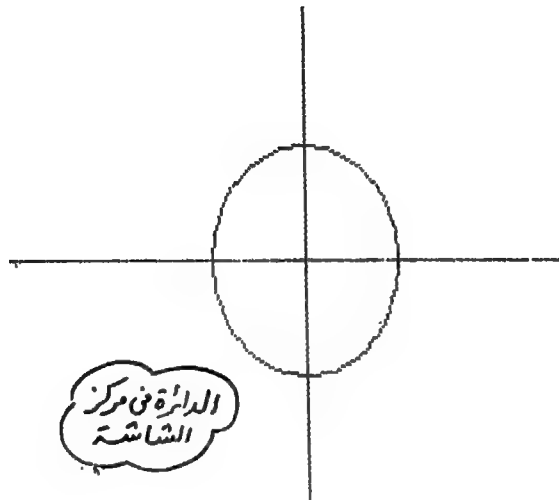
كانت الدائرة على اليمين (أى تستخدم الأمر RT) لزم إزاحة السلحفاه جهة اليسار . ولو كانت الدائرة على اليسار لزم إزاحة السلحفاه جهة اليمين . ولو كانت الدائرة إلى أعلى لزم إزاحتها إلى أسفل وهكذا .. أما مسافة الإزاحة فهي تساوى نصف القطر R .

والشكل التالى يوضح خطوات تحريك السلحفاه إلى أعلى ثم إدارتها إلى اليمين لترسم دائرة (إلى أسفل) وفى النهاية يتم تحريك السلحفاه إلى بيتها الأصلي فترى الدائرة فى المنتصف تماماً .



رسم دائرة فى منتصف الشاشة

والكمبيوتر IBM يمد بوسيلة جاهزة لهذا الغرض وهى البرنامج الفرعى CCIRCLE الموضح فى الملحق (٢) فى نهاية الكتاب .



ويمكننا استخدام البرنامج CCIRCLE كما هو أو لإنشاء برنامج آخر بأنفسنا يؤدي نفس الغرض سواء كان يحمل نفس الاسم أو اسماً مختلفاً .

والبرنامج التالي يقوم برسم الدائرة في منتصف الشاشة كما هو موضح بالرسم وقد أطلقنا عليه الاسم "C" وهو يأخذ مدخلين A , R : ومن الممكن بالطبع تحديد القيمة (A) والاكتفاء بالمتغير R الذي يمثل نصف القطر وبذلك يكون مطابقاً للبرنامج CCIRCLE للكمبيوتر IBM .

وهذا هو البرنامج :

برنامج فرعي لرسم دائرة في منتصف الشاشة
?
TO C : R : A
MAKE "S 2 * 3.14159 * : R / : A
PU FD : R RT 90 PD
REPEAT : A [RT 360 / : A / 2 FD : S
RT 360 / : A / 2]
LT 90 PU BK : R PD
END

ولتبسيط البرنامج أيضاً يمكن التعويض عن عدد الأجزاء (الأضلاع) بالعدد ٣٦ فنحصل على البرنامج في الصورة التالية :

```

?
TO C36 :R
MAKE "S 3.14159 * :R / 18
PU FD :R RT 90 PD
REPEAT 36 [RT 5 FD :S RT 5]
LT 90 PU BK :R PD
END

```

لرسم دائرة في منتصف الشاشة مع 36 ضلعاً

ولمقارنة هذا البرنامج بالبرنامج السابق يتم تشغيلهما واحداً وراء الآخر على شاشة واحدة فنحصل على نفس الدائرة .

أظهر السلحفاه أولاً ثم أعط أمر تشغيل البرنامج الأول :

C 40 36

وبعد رسم الدائرة أعط أمر تشغيل البرنامج الثاني :

C36 40

سوف نجد أنهما متطابقان .

4 رسم مجموعة دوائر ذات مركز واحد :

البرنامج التالي لرسم مجموعة من الدوائر المتمركزة باستخدام البرنامج 36 كبرنامج فرعى .

وهذا هو البرنامج مصحوباً بالتنفيذ .



دوائر متمركزة

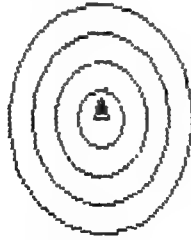
```

?
TO CN
C36 10 C36 20 C36 30 C36 40
END

```

والشكل يوضح مجموعة من الدوائر ذات أنصاف أقطار 10 ، 20 ، 30 ، 40 ، وهي مرادفة للأوامر التي يحتوى عليها البرنامج .

فالأمر C36 10 يختص برسم الدائرة ذات نصف القطر 10 .
والأمر C36 20 يختص برسم الدائرة ذات نصف القطر 20 وهكذا ..
ويمكن رسم نفس المجموعة باستخدام البرنامج C كالآتي :



```
TO CON
C 10 36 C 20 40 C 30 50 C 40 60
END
```

ونلاحظ في البرنامج استخدام أرقام مختلفة للدليز الثاني وهي لا تتغير النتيجة من شيء إلا في درجة الدقة وسرعة التنفيذ .

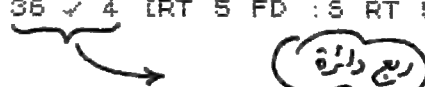
٥ الأشكال الحلزونية :

أما هذا البرنامج فيرسم أشكالاً كالحلزون الموضح بعد :



وهو يتكون من روتين فرعى لرسم قوس نصف قطره متغير (R) وهذا البرنامج هو نفسه البرنامج C36 مع تخفيض عدد مرات التكرار إلى الربع .

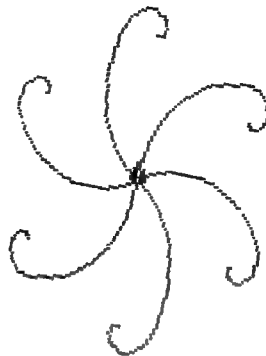
```
?
TO ARC36 :R
MAKE "S 3.141592 * :R / 18
REPEAT 36 ✓ 4 [RT 5 FD :S RT 5]
END
```



ويعمل هذا البرنامج الفرعى بإعطاء الأمر ARC36 يعقبه رقم ما يمثل نصف قطر الدائرة (القوس جزء من دائرة) . وبتكرار تنفيذ هذا البرنامج مرة بعد مرة مع تصغير قيمة نصف القطر كل مرة نحصل أشكال حلزونية مختلفة .

وهذا البرنامج مكافئ تماماً للبرنامج الفرعى RARC للكمبيوتر IBM (راجع الملحق (٢)) .

ويمكن استخدامه لرسم شكل أخطبوطى ذى ستة أرجل كالوضح بعد .



```
?
TO SPIRAL
ARC36 40 ARC36 10
ARC36 5 ARC36 2
END
```

```
?
TO SPS
MAKE "Z 60
REPEAT 6 [SPIRAL PU HOME PD LT
Z MAKE "Z :Z + 60]
END
```

وهنا يقوم البرنامج الفرعى الأول (SPIRAL) برسم رجل واحدة من أرجل الأخطبوط ثم يتولى البرنامج الرئيسى SPS تكرار العملية .

والملاحظة التى تجدد علينا هنا هى العدّاد الذى أنشأناه فى البرنامج SPS . فالبرنامج يبدأ بتخصيص 60 للمتغير S الذى يستخدم كزاوية لإدارة الحلزون SPIRAL بعد رسم كل رجل . ونظراً لأن السلحفاة تعود لمنزلها بعد كل حلزون فيلزم تعديل اتجاهها كل مرة بزاوية تزيد عن سابقتها بمقدار ٦٠ درجة لذلك جاءت العبارة :

MAKE "Z :Z+60

بمعنى : « أضف إلى قيمة المتغير (Z) مقدار (60) وسمّ الناتج (Z) » .

● ضبط الدوائر على الشاشة SCRUNCH, SETSCRUNCH

ذكرنا من قبل أن الدائرة ربما تبدو على الشاشة كما لو كانت قطعاً ناقصاً ellipse بمعنى أن قطرها فى المحور الأفقى يختلف عنه فى المحور الرأسى . وهذا ناتج عن اختلاف طول خطوة السلحفاة فى الاتجاه الأفقى عنه فى الاتجاه الرأسى والتى نطلق عليها نسبة الواجهة (aspect ratio) .

والشاشة القياسية فى لغة لوجو واجهتها تعطى بالنسبة :

[100 100]

ويمكن الحصول على هذه النسبة بالأمر الآتى :

? PRINT SCRUNCH

[100 100]

ويمكن التحكم فى هذه النسبة بالأمر SETSCRUNCH .

وبتغيير هذه النسبة يتغير طول خطوة السلحفاة فى أحد الاتجاهين أو كليهما . ففى البرنامج التالى سبق رسم الدائرة ضبط النسبة لتكون [100 50] فنتج عن ذلك رسم القطع الناقص الموضح :



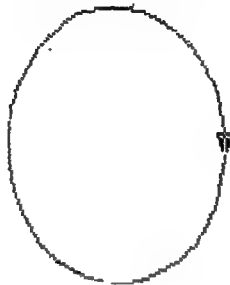
```
?
TO C
SETSCRUNCH [100 50].
REPEAT 36 [fd 50 * sin 10 rt 10]
END
```

فإذا كانت الدائرة أصلاً غير متماثلة على شاشتك فإمكانك استخدام هذا الأمر لتعديل أحد رقمي النسبة بحيث تظهر الدائرة متماثلة . كما يمكن استخدام هذا الأمر أيضاً لضبط المربعات التي تظهر مستطيلة على الشاشة .

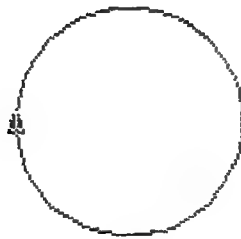
وقد رأينا قبل أن نسبة الواجهة قد تؤثر على شكل الرسم المطبوع أيضاً بحيث تظهر الدائرة سليمة على الشاشة ومنبعجة على الورق .

في هذه الحالة يتم تعديل النسبة بحيث يُضبط تماثل الرسم على الورق حتى لو بدا غير متماثل على الشاشة .

والمثال الآتي يوضح كيف أن النسبة [100 100] تؤدي إلى طبع قطع ناقص بينما النسبة [100 80] تؤدي إلى طبع دائرة .



```
?
TO SC
SETSCRUNCH [100 100]
REPEAT 36 [FD 50 * SIN 10 RT 10]
END
```



```
?  
TO SC  
SETSCRUNCH [100 80]  
REPEAT 36 [FD 50 * SIN 10 RT 10]  
END
```

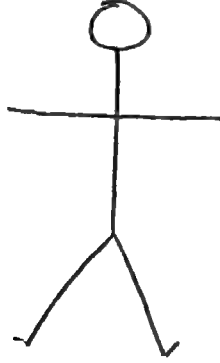
وبتغيير رقمى الواجهة معاً يمكن تكبير وتصغير الرسم بالكامل . جرب
[50 50] , [150 150] وشاهد النتيجة .

٧

مشروعات للرسم

● لترسم شخصاً :

الخطوة الأولى التي نتخذها عندما نريد أن ننفذ رسماً بالكمبيوتر أن نرسمه رسماً سريعاً باليد « مسودة » حتى نعرف الرسم المطلوب بشيء من التفصيل .
وهذا هو رسم اليد :



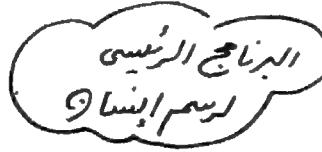
هذا الرسم مكون من رأس وذراعين ورجلين وجسم ولا بأس من إضافة غطاء للرأس أيضاً (قبعة مثلاً) .

إذن فالبرنامج الرئيسى الذى سنكتبه سيحتوى على البرامج الفرعية التى ترسم كل جزء، على حدة . ولنبدأ بكتابة هذا البرنامج الرئيسى PERSON الذى لا يحتوى على أوامر لوجو القياسية وإنما يحتوى على أسماء البرامج الفرعية التى ستقوم بالعمل وهى :

البرنامج BODY لرسم الجسم والبرنامج LEGS للأرجل و ARMS للذراعين و HEAD للرأس و HAT للقبعة .

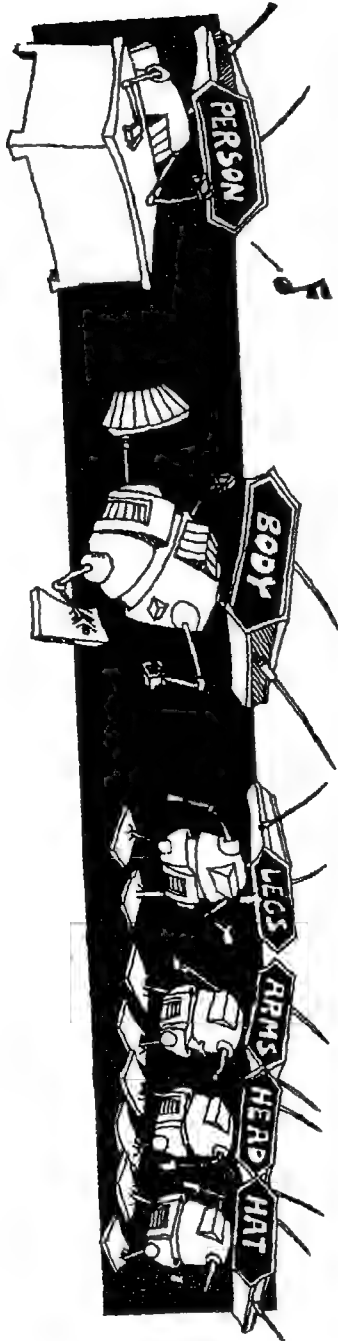
وهذا النوع من البرامج الرئيسية قد أسميناه قبلاً Superprocedure لأن يرأس جميع البرامج

TO PERSON
BODY
LEGS
ARMS
HEAD
HAT
END



البرنامج الرئيسى

وبذلك فإن ما يحدث بداخل الكمبيوتر لا يخرج كثيراً عن التصور
الموضح في الشكل التالى حيث يجلس الروبوت الرئيس PERSON إلى مكتبة
مصدراً الأوامر للروبوتات المساعدة الأخرى والتي تحمل أسماء أجزاء الجسم .
ولا يفوتنا أن نلاحظ أن كلاً من الرئيس والمرعوسين عبارة عن روبوتات
مبتكرة ! (الشكل السادس للشاشة) .



ما يحدث بداخل الكمبيوتر عند تنفيذ البرنامج PERSON

ولا بأس من كتابة البرنامج الرئيسي قبل البرامج الفرعية ولكنه مع ذلك برنامج غير قابل للتنفيذ بعد . لأن الكمبيوتر لا يعرف ما هو BODY وما هو ARMS وإلى آخره . ولو أننا حاولنا تنفيذ البرنامج PERSON الآن سوف يشكو الكمبيوتر كالمثال الآتي :

? PERSON

I don't know how to BODY in PERSON

لقد توقف عند أول أمر في البرنامج وهو BODY .

● فلنبدأ بتعليم الكمبيوتر كيف ينفذ الأمر BODY :

TO BODY
FORWARD 30
BACK 30
END



إن الجسم المطلوب مجرد خط واحد يعقبه إعادة السلحفاه إلى مكانها بالأمر .BACK

جرب تنفيذ البرنامج الآن . سوف تحصل على الشكل المرسوم ثم يتوقف الكمبيوتر عند الأمر الثاني وهو LEGS مرسل الشكوى كالمثال الآتي :

I don't know how to LEGS in PERSON

● فلنعلمه كيف ينفذ البرنامج LEGS :

TO LEGS
LEFTLEG
RIGHTLEG
END



ولكن البرنامج LEGS الممثل للأرجل يحتوى على الرجل اليسرى LEFTLEG والرجل اليمنى RIGHTLEG وكلاهما أمران جديداً أيضاً :
هذا البرنامج يحتاج لبرنامجين فرعيين يرأسهما :

TO LEFTLEG
RIGHT 30
BACK 30
LEFT 90
FORWARD 5
BACK 5
RIGHT 90
FORWARD 30
LEFT 30
END

تعادة السلحفاه إلى بيتها

الرجل اليسرى



والأوامر الأربعة الأخيرة تعيد السلحفاه إلى بيتها ومن الممكن استبدالها بالأوامر التالية : PU HOME PD

● أما البرنامج الخاص بالرجل اليمنى فهو مماثل تماماً للبرنامج السابق مع تبديل كل انحراف جهة اليمين (RT) بنظيره جهة اليسار (LT) والعكس بالعكس .

TO RIGHTLEG
LEFT 30
BACK 30
RIGHT 90
FORWARD 5
BACK 5
LEFT 90
FORWARD 30
RIGHT 30
END

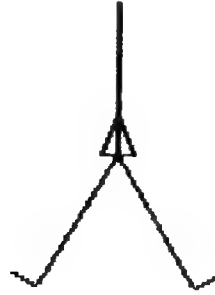
الرجل اليمنى

عند تنفيذ هذا البرنامج نحصل على الرسم التالي :



هل نجرب البرنامج الرئيسي PERSON ؟

سوف نحصل على الرسم التالي قبل أن يشكو الكمبيوتر من عدم استطاعته تنفيذ الأمر ARMS .



● والآن فلنعلم الكمبيوتر كيف يرسم الذراعين بالبرنامج ARMS :

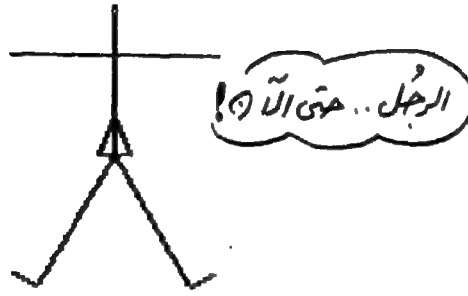
TO ARMS
FORWARD 20
RIGHT 90
FORWARD 20
BACK 40
FORWARD 20
LEFT 90
BACK 20
END

إعادة السطوف إلى بيتها

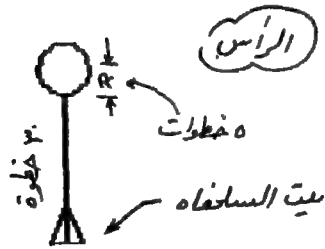
الذراع



حتى الآن قد حصلنا على هذه الأجزاء الموضحة من جسم الرجل :



● لنكتب الآن برنامج الرأس : HEAD



يمكن هنا أن نستعين بالبرنامج الفرعي (C3) الذي استخدمناه من قبل لرسم الدائرة أو أى برنامج بسيط لرسم الدائرة . أما مستخدمو الكمبيوتر IBM فيمكنهم استخدام البرامج الفرعية الجاهزة الخاصة بالدوائر والأقواس على القرص (LWIL) وسوف نستخدم القيمة 5 كنصف قطر للدائرة التي تمثل الرأس باعتبارها قيمة مناسبة . وهذا هو برنامج الرأس HEAD .

```
?
TO HEAD
PU FD 30
LT 90 PD
C3 5
PU HOME PD
END
```

الرأس

رسم الدائرة بالبرنامج C3

في هذا البرنامج يتم رفع القلم تمهيدا للحركة دون رسم . تتحرك السلاحف إلى أعلى الجسم (FD 30) ثم تدار إلى اليسار ٩٠ درجة مع خفض القلم .

عندئذ يبدأ البرنامج الفرعي (C3) في رسم دائرة نصف قطرها 5 وبعد الانتهاء يُرفع القلم مرة أخرى وتعود السلاحف إلى بيتها ثم يُخفض القلم تمهيدا للرسم التالي .

والبرنامج الفرعى (C3) كآلاتى :

برنامج الدائرة C3

```
?  
TO C3 :R  
MAKE "S 3.141592 * :R / 18  
REPEAT 36 [RT 5 FD :S RT 5]  
END
```

بعد رسم الرأس تكون السلحفاه فى بيتها ورأسها متجه إلى أعلى .

بهذا يكون شكل الرجل قد اكتمل ولا بأس من أن نمنحه قبعة يرتديها
كنوع من التدريب على الرسم .

وهذا هو البرنامج HAT لرسم القبعة :

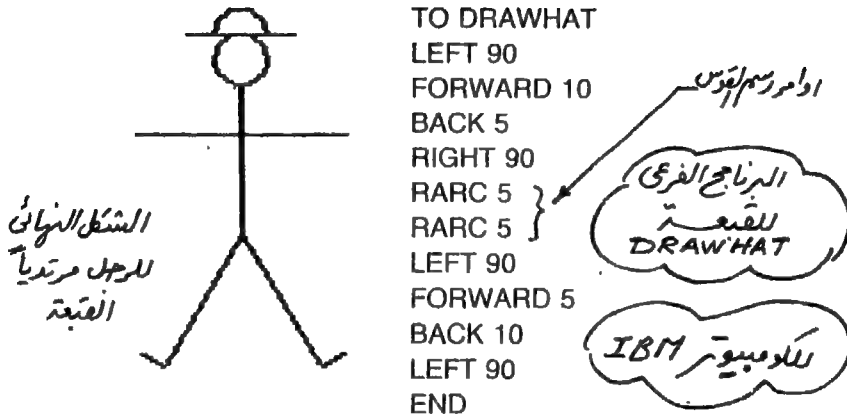
```
TO HAT  
PENUP  
FORWARD 40  
PENDOWN  
DRAWHAT  
PENUP  
BACK 40  
END
```

برنامج القبعة

برنامج فرعى

ويشمل البرنامج التحضيرات الأساسية لرسم القبعة وهى الحركة إلى أعلى
الرأس (FD 40) والعودة بعد الرسم إلى مركز الشاشة (BK 40) ، أما القبعة
نفسها فيتم رسمها بالبرنامج الفرعى DRAWHAT .

والبرنامج التالى هو البرنامج الفرعى لرسم القبعة وهو خاص بالكمبيوتر
IBM حيث يستخدم البرامج الفرعية الخاصة RARC . والشكل النهائى للرجل
مرتدياً القبعة موضح مع البرنامج .



فماذا بالنسبة لباقي أجهزة الكمبيوتر ؟ يمكننا ببساطة أن ننشئ برنامجاً فرعياً لرسم القوس الذي تتكون منه القبعة ولا مانع من أن نطلق عليه نفس الاسم RARC وفي هذه الحالة يمكن استخدام برنامج IBM مع أي كومبيوتر آخر .

هل نتذكر البرنامج ARC36 الذي قدمناه من قبل لرسم القوس ؟
سوف نستخدمه الآن ونمنحه اسماً جديداً هو RARC حتى يكون مطابقاً لبرنامج IBM . وهذا هو البرنامج .

```
?
TO RARC :R
MAKE "S 3.141592 * :R / 18
REPEAT 36 / 4 [RT 5 FD :S RT 5]
END
```

البرنامج الفرعي RARC

فلاش



البرنامج السابق يبدأ بالأمر
MAKE الذى يحدد قيمة المتغير S .

ماذا تعتقد لو حذفنا هذا السطر
من البرنامج ؟

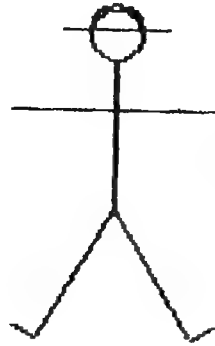
في الحقيقة أنه لن يتغير شيء وسوف يُرسم الشخص دون أى شكوى
من لوجو .

لماذا ؟ ... لقد أشرنا من قبل أن هناك متغيرات عامة يتم اختزانها في
حيز العمل (workspace) عموماً ومتغيرات خاصة بالدوال يقتصر
استخدامها على برنامج الدالة فقط . والمتغير S من النوع العام ويكفى
تعريفه مرة واحدة في أى برنامج من البرامج .

تجربة



هل تستطيع تعديل أحد البرامج
الفرعية السابقة لضبط وضع القبعة
كما في الرسم التالي ؟



قبعة
النزاهة

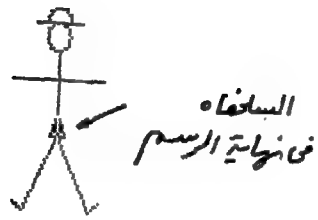
فكرة

إذا حاولت رسم الشخص مرتين متتبعين فسوف تلاحظ أنه في المرة التالية سيختفى جزء من جسم الرجل . علل لذلك . وإذا احتجت لمعاونة يمكنك مقارنة البرنامج التالي بنظيره من البرامج التي عرضناها من قبل .



```
?  
TO HAT  
PU FD 40  
PD DRAWHAT  
PU BK 40 PD  
END
```

. وبمناسبة الحديث عن حيز العمل وما يحتويه من برامج ، نقدم هنا طبعة كاملة لحيز العمل بما يحتويه من برامج ومتغيرات أثناء رسم الرجل .
والآتي بعد لائحة بأسماء البرامج المستخدمة مع الشكل النهائي للرجل .



```

?
TO RARC :R
TO C3 :R
TO DRAWHAT
TO LEFTLEG
TO RIGHTLEG
TO HAT
TO HEAD
TO ARMS
TO LEGS
TO BODY
TO PERSON

```

وفيما يلي نص البرامج التي استخدمت علاوة على المتغيرات التي احتوى عليها حيز العمل بالكامل .

?
 TO RARC :R
 MAKE "S 3.141592 * .R .18
 REPEAT 36 / 4 [RT 5 FD .5 RT 5]
 END

TO C3 :R
 MAKE "S 3.141592 * .R .18
 REPEAT 36 [RT 5 FD :3 RT 5]
 END

TO DRAWHAT
 LT 90 FD 10
 BK 5 RT 90
 RARC 5 RARC 5
 LT 90 FD 5 BK 10 LT 90
 END

TO LEFTLEG
 RT 30 BK 30 LT 90
 FD 5 BK 5 RT 90
 FD 30 LT 30
 END

TO RIGHTLEG
 LT 30 BK 30 RT 90 FD 5
 PU HOME PD
 END

TO HAT
 PU FD 40
 PD DRAWHAT
 PU BK 40 PD
 END

البرامج والتغيرات
 في خيز الحمار

TO HEAD
 PU FD 30
 LT 90 PD
 C3 5
 PU HOME PD
 END

TO ARMS
 FD 20 RT 90 FD 20
 BK 40
 FD 20 LT 90 BK 20
 END

TO LEGS
 LEFTLEG
 RIGHTLEG
 END

TO BODY
 FD 30
 BK 30
 END

TO PERSON
 BOD
 LEGS
 ARMS
 HEAD
 HAT
 END

المتغير الثاني S

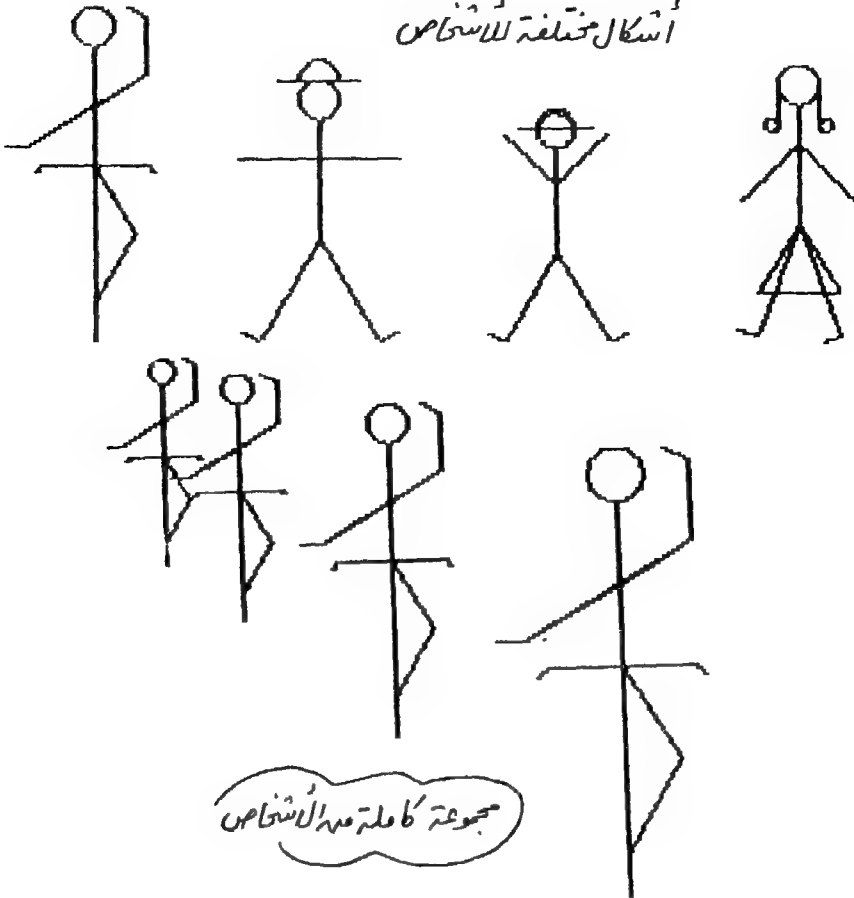
MAKE "S '0.27266444

تمارين



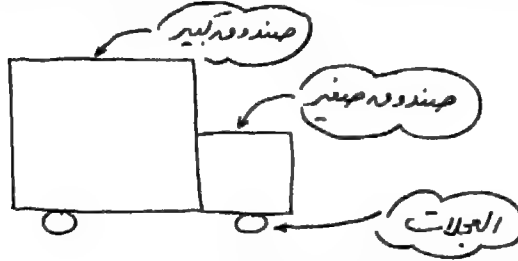
باستخدام البرنامج السابق حاول
إجراء التعديلات المناسبة للحصول
على الأشكال الآتية :

أشكال مختلفة للأشخاص



● قافلة من السيارات :

لنبدأ بمسودة للرسم باليد نستوضح فيها معالم السيارة المطلوبة :



بهذا نحتاج إلى ثلاثة برامج فرعية كالاتي :

TO TRUCK
BIGBOX ← الصندوق الكبير
SMALLBOX ← الصندوق الصغير
WHEELS ← العجلات
END

البرنامج الرئيسي للسيارة

فلننشئ الآن البرامج الفرعية واحداً تلو الآخر .

TO BIGBOX ← الصندوق الكبير
REPEAT 4 [FORWARD 60 RIGHT 90]
END

TO SMALLBOX ← الصندوق الصغير
REPEAT 4 [FORWARD 30 RIGHT 90]
END

TO WHEELS ← العجلات
RIGHT 90
RCIRCLE 5
FORWARD 90
RCIRCLE 5
BACK 90
LEFT 90
END

IBM*

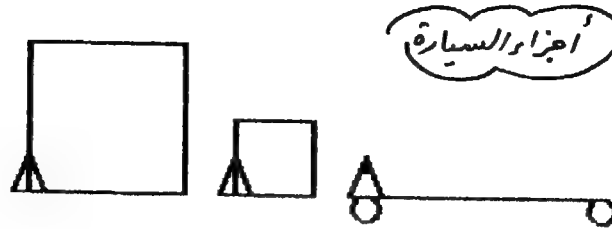
ولا يفوتنا في البرنامج الفرعى الأخير ظهور العبارات RCIRCLE التى يختص بها الكمبيوتر IBM وعلينا إنشاء ما يناظرها من برامج فرعية لرسم دائرة نصف قطرها 5 عند استخدام كومبيوتر آخر .

والبرنامج الفرعى CIRCLE يصلح لهذا الغرض حيث أنه يرسم دائرة ذات نصف قطر 5 . ومع ذلك فمن المفضل إنشاء برنامج عام لرسم دائرة بأى نصف قطر مطلوب باستخدام الدوال كالمثال الآتى :

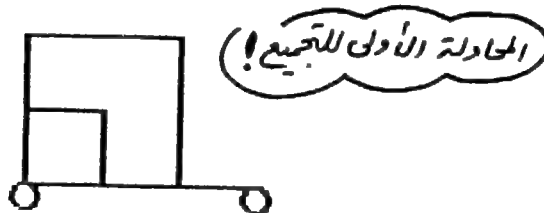
```
?
TO CIRC :R
MAKE "S 360 / :R / 2 / 3.1415927
REPEAT 360 / :S [FD 1. RT :S]
END
```

برنامج فرعى لرسم دائرة نصف قطرها R

ويستخدم هذا البرنامج بكتابة اسمه (CIRG) متبوعاً بنصف القطر المطلوب . وبناء هذه البرامج الفرعية يمكننا الحصول الأشكال التالية التى تتكون منها السيارة .

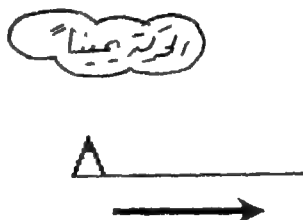


وإذا نفذنا البرنامج (TRUCK) الآن فسوف نحصل على الشكل التالى :

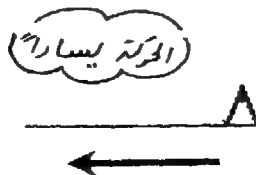


لا تزال السيارة في حاجة إلى تعديل حتى نحصل على الشكل المطلوب .
 فيلزمنا تحريك الأجزاء يميناً ويساراً حتى تتم عملية التجميع على الوجه المراد .
 وهذا يتطلب بناء برنامجين فرعيين آخرين الأول للحركة يميناً وليكن اسمه
 MOVEOVER والثاني للحركة يساراً وليكن اسمه MOVEBACK .

TO MOVEOVER
 RIGHT 90
 FORWARD 60
 LEFT 90
 END



TO MOVEBACK
 LEFT 90
 FORWARD 60
 RIGHT 90
 END

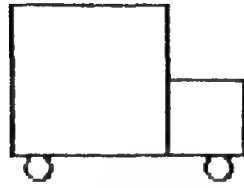


وبإدخال هذه التعديلات على البرنامج الرئيسى يصبح كالتالى :

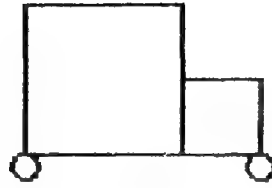
TO TRUCK
 BIGBOX
 MOVEOVER
 SMALLBOX
 MOVEBACK
 WHEELS



وبتنفيذ البرنامج TRUCK الآن فإنه يعطى الصورة (أ) الموضحة في الشكل
 التالى وهى تختلف قليلاً عن الشكل الأصلي الذى رسمناه باليد من ناحية مكان
 العجلات . فإذا أردت الحصول (ب) المطابقة للشكل الأصلي فعليك بإجراء
 التعديل اللازم بالبرنامج .



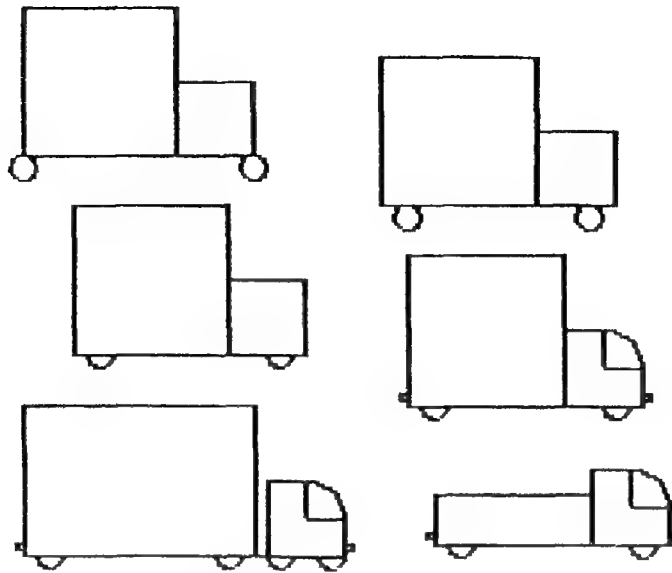
(ب)



(p)

هذه السيارة هي الخلية الأولى التي نبني منها قافلة السيارات سواء بالتكرار أو بإضافة التعديلات إلى شكل السيارة لكي تشمل القافلة على سيارات النقل « ونصف النقل » بمختلف الأحجام والأشكال .

والشكل التالي يوضح بعض هذه النماذج التي يمكن برمجتها بسهولة .

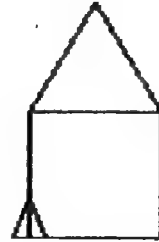




● منطقة سكنية :

لنبدأ بمنزل واحد بالطريقة التقليدية . المنزل يتكون من مربع ومثلث .
وكل من المربع والمثلث يمكن برمجته على حدة كبرنامج فرعى صغير باستخدام
ضلع طوله ٥٠ خطوة .

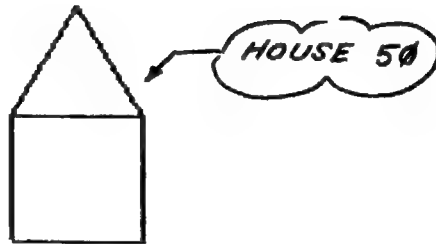
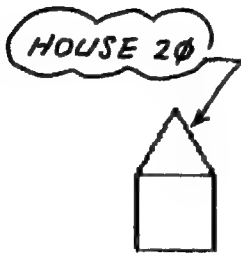
TO HOUSE
SQUARE → برنامج المربع
FORWARD 50
RIGHT 30
TRIANGLE → برنامج المثلث
LEFT 30
BACK 50
END



نتيجة تنفيذ
البرنامج

ويمكن التحكم في حجم المنزل انا نتج باستخدام متغير SIZE لتحديد حجم
المربع والمثلث معاً . والمتغير SIZE يمثل طول الضلع لأي من الشكلين فيصبح
البرنامج كالاتي :

TO HOUSE :SIZE
SQUARE :SIZE
FORWARD :SIZE
RIGHT 30
TRIANGLE :SIZE
LEFT 30
BACK :SIZE
END



وكما نرى أنه يمكن الحصول على أحجام مختلفة من المنازل بتحويل البرنامج إلى برنامج للدالة بدلاً من الطريقة العادية والرسم يوضح منزلين أحدهما تم تنفيذه بالأمر .

? HOUSE 20

والآخر باستخدام الأمر :

? HOUSE 50

وللتوسع في بناء المنازل يلزمنا روتين لتحريك الشكل يمينا أو يساراً على غرار الروتين MOVEOVER أو الروتين MOVEBACK .

ولتعميم الروتين يمكن إضافة تعديلين إلى البرنامج MOVEOVER .

أولاً : استخدام الدوال في برنامج تحريك الأشكال حتى يكون عاماً بحيث نستخدمه في أى برنامج .

ثانياً : نكتفى باستخدام روتين واحد وليكن MOVEOVER بحيث يستقبل قيمة موجبة للمتغير عند الحركة يمينا وقيمة سالبة عند الحركة إلى اليسار . وهذا هو البرنامج الفرعى :

```
TO MOVEOVER :SIZE
PENUP
RIGHT 90
FORWARD :SIZE + 10
LEFT 90
PENDOWN
END
```

البرنامج الفرعى للحريك

لاحظ إضافة 10

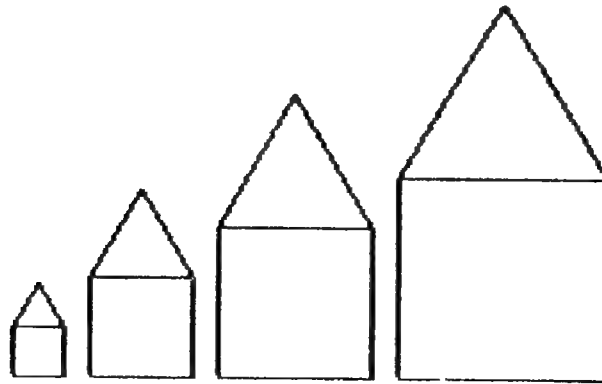
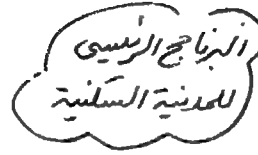
ولا يفوتنا ملاحظة إضافة 10 على المتغير SIZE مما ينتج عنه « انخياز » إلى اليمين قدره عشر خطوات . فإذا أدخلنا الرقم 60 للبرنامج تم الإزاحة بمقدار 70 ، وإذا أدخلنا الرقم 60— تم الإزاحة بمقدار (50 -) .

أما برنامج المساكن فهو كالآتي :


```

TO HOUSES
MOVEOVER -100
HOUSE 10
MOVEOVER 10
HOUSE 20
MOVEOVER 20
HOUSE 30
MOVEOVER 30
HOUSE 40
MOVEOVER 40
END

```



وهناك مع ذلك طريقة بديلة مختصرة وهي بناء البرنامج الرئيسي نفسه كدالة باستخدام متغير SIZE أيضاً كالآتي :

```

TO GROWHOUSES :SIZE
HOUSE :SIZE
MOVEOVER :SIZE
GROWHOUSES :SIZE + 10
END

```



وهذا البرنامج يزيد حجم المنزل كل مرة بإضافة 10 إلى المتغير SIZE .

والصفة الجديدة التي يختص بها هذا البرنامج عن سابقة أنه لن يتوقف إلا إذا تدخلت من الخارج بالضغط على الزر (ctrl-beak) أو ما يعادله في الأجهزة المختلفة .

ولو تركت البرنامج يعمل لفترة طويلة فإنك تحصل على نتيجة مؤسفة إذ
تتراكم المنازل فوق بعضها البعض حتى تسود الشاشة تماماً من كثرة الخطوط
المتزاحمة بها .

وسنرى في الأجزاء التالية كيف نوقف مثل هذا البرنامج من داخل البرنامج
نفسه .

٨

التكرار .. واتخاذ القرار

● الحلقات التكرارية :

أحياناً يدور الإنسان في حلقة مفرغة لا يستطيع الخلاص منها إلا بمعجزة أو بتدخل حاسم من الأهل أو الأصدقاء المخلصين .

وهذا الموقف كثيراً ما يتعرض له الساحر لوجو عند تنفيذه لأمر ما فيجد نفسه واقعاً في حلقة تكرارية بلا نهاية لا يستطيع الخروج منها .

وقد تصادف السلحفاه أيضاً هذا الموقف وهى ترسم على الشاشة دون أن ينتهى عملها كما حدث فى برنامج المنطقة السكنية .

ولا بد فى هذه الحالة من الاستعانة « بنظام التشغيل » للكمبيوتر الذى يسعفنا بالضغط على زر معين فيوقف أى برنامج جارى تنفيذه ..

والمثال الآتى يوضح نوعية من هذه الحلقات التكرارية اللانهائية حيث يصاب أحد الروتينات الفرعية (LOOP) بنوبة من جنون العظمة فيستمر فى القول « أنا رجل عظيم .. I am a great man » بلا نهاية . ولا يتوقف هذا البرنامج الفرعى (LOOP) إلا عندما نضغط على زر الإيقاف الاضطرارى فيتوقف مُجبراً ..

?
TO LOOP
PR [,...I AM A GREAT MAN]
LOOP
END

البرنامج الفرعي LOOP

- تنفيذ البرنامج -

...I AM A GREAT MAN
...I AM A GREAT MAN
...I AM A GREAT MAN
...I AM A GREAT MAN
...I AM A GREAT MAN
...I AM A GREAT MAN
...I AM A GREAT MAN
...I AM A GREAT MAN
...I AM A GREAT MAN
...I AM A GREAT MAN
...I AM A GREAT MAN
...I AM A GREAT MAN
...I AM A GREAT MAN
...I AM A GREAT MAN
...I AM A GREAT MAN
...I AM A GREAT
STOPPED!!! in LOOP

إيقاف البرنامج
نتيجة الخارج

IF , STOP

● كيف نوقف الحلقة التكرارية

لعلنا لاحظنا في البرنامج السابق أن البرنامج لا يصل إلى نهايته أبداً فكلما طبع القائمة I am a great man استدعى البرنامج نفسه من جديد بالأمر LOOP فيكرر ما يفعل .

ولكي نوقف مثل هذا البرنامج لا بد من أن نضمنه خاصية القدرة على اتخاذ القرار . ولنفرض أننا نريد تكرار الحلقة خمس مرات فقط . هذا معناه أن يجب أن يحتوي البرنامج على عدّاد ليعدّ من واحد إلى خمسة وبجانب ذلك يجب أن يكون هناك من يراقب العدّاد فإذا وصل إلى الرقم خمسة تدخل لإنهاء العمل وإيقاف البرنامج .

لا شك أننا سنحتاج لتعلم بعض الأوامر الجديدة من أوامر لوجو .

أنظر هذا التعديل في البرنامج :

العداد

```

TO LOOP :I
PR [...I AM A GREAT MAN]
IF :I = 5 [STOP]
LOOP :I + 1
END

```

الشرط...والقرار

```

...I AM A GREAT MAN
...I AM A GREAT MAN
...I AM A GREAT MAN
...I AM A GREAT MAN
...I AM A GREAT MAN

```

لقد استخدمنا الدليل I في مستهل البرنامج كعداد .

وبعد طبع كل قائمة يزداد العداد بمقدار واحد كما في السطر الأخير :

LOOP :I + 1

أى أن البرنامج عندما يستدعى نفسه للتكرار فإنه يزداد قيمة الدليل بمقدار 1-
.. هذا هو العداد ..

أما القرار فيأتى بعد طبع القائمة مباشرة بموجب العبارة الجديدة :

IF :I = 5 [STOP]

تقول هذه العبارة : « إذا كان العداد I مساوياً للرقم 5 فننفذ الأمر :
STOP . وإلا فانتقل إلى الأمر التالى للأمر STOP فإذا تحقق الشرط في الدورة
الخامسة نفذ البرنامج الأمر STOP وخرج من الحلقة . ويمكن فهم العملية
الشرطية بصورة عامة كالآتي :

القائمة الثانية [القائمة الأولى] الشرط IF

فإذا تحقق الشرط أيأ كان هذا الشرط فإن قائمة الأوامر يتم تنفيذها أما إذا لم
يتحقق الشرط فإن التنفيذ يتجه إلى القائمة الثانية . ونلاحظ أن القائمة الأولى
توضع بين القوسين المربعين لفصلها عما يليها من أوامر .

والقائمة الأولى هنا تحتوى على أمر واحد هو STOP ولكن يجوز أن تشمل
أكثر من أمر .

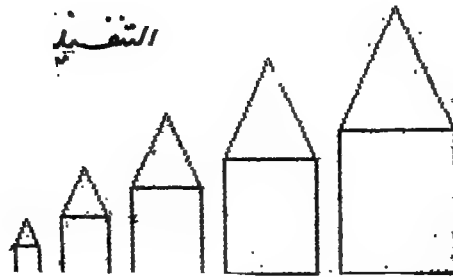
● ويمكن وضع شرط أيضاً في برنامج المساكن بنفس الطريقة فلو أردنا مثلاً ألا يزيد حجم أكبر منزل عن ٥٠ فإن الشرط يمكن أن يتحكم في الحجم SIZE كالآتي :

```
?
TO JOB
PU BK 50 PD
MOVE -100
HOUSESGR 10
END
```

← البرنامج الرئيسي

```
?
TO HOUSESGR :S
IF :S > 50 [STOP]
HOUSE :S MOVE :S
HOUSESGR :S + 10
END
```

← العدد الفرعي
← الشرط والقرار



والبرنامج الذي يقوم ببناء المنازل هو البرنامج الفرعي S: HOUSESGR والمتغير S هو نفسه متغير الحجم SIZE حيث يتحكم في حجم المنازل بزيادته بمقدار 10 عند كل رسم جديد .

أمال البرامج الفرعية المستخدمة بداخل هذا البرنامج فهي MOVE وهو ذاته البرنامج MOVEOVER الذي استخدمناه من قبل .

والشرط الذي وضعناه في البرنامج يقول :

« إذا زاد حجم المنزل عن 50 توقف »

أما البرنامج الرئيسى JOB فهو يمهّد للبرنامج HOUSESGR بتحريك
السلحفاه إلى مكان بداية مناسب باستخدام البرنامج الفرعى MOVE .

فلاش



ما الفرق بين التكرار باستخدام
الأمر REPEAT وبين استخدام
الحلقات التكرارية ذات الشرط التى
تحدثنا عنها مؤخراً ؟

إن الفارق الرئيسى هو أن الأمر
REPEAT يفيد فى تكرار عملية
معينة عدداً معيناً من المرات .

أما استخدام الشرط فهو أكثر عمومية حيث يمكن أن يتحكم الشرط
فى عدد المرات ، أو فى أى بيان آخر بصرف النظر عن عدد المرات . وفى
المثال الأخير يتحكم الشرط فى حجم المنزل . وفى الأمثلة القادمة سوف
نلتقى بالعملية الشرطية وهى تتحكم فى بيانات مختلفة تماماً بدون تحديد
مسبق لعدد المرات .

HEADING

● التحكم فى وضع السلحفاه

اكتب هذا البرنامج الذى يستدعى البرنامج الفرعى SQUARE (وهو برنامج
رسم المربع) ويديره بزاوية معينة خلال حلقة تكرارية لا نهائية .

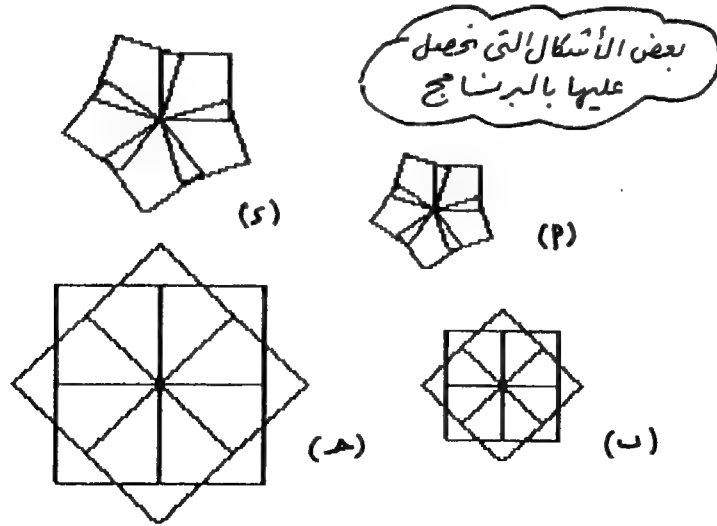
وهو يعمل بإدخال رقمى طول الضلع والزاوية مثل :

? SPINSQUARES2 50 45

```

TO SPINSQUARES2 :SIZE :ANGLE
SQUARE :SIZE
RIGHT :ANGLE
SPINSQUARES2 :SIZE :ANGLE
END

```



مثل هذا البرنامج يصعب إيقافه بتحديد عدد مرات التكرار للمربع الدائر من ناحية لأن هذا العدد يتغير مع كل زاوية مُدخلة ، فمع الزاوية ٩٠ درجة نحصل على ٤ مربعات ومع الزاوية ٤٥ درجة ثمانية مربعات كما في الشكل (ج) ومن ناحية أخرى فإنه من الصعب التنبؤ مع بعض الزوايا بعدد مرات التكرار اللازمة لرسم الشكل النهائي المتماثل .

لكن هناك صفة مشتركة في جميع الأشكال التي نحصل عليها من البرنامج وهي أن الشكل يكتمل عندما تعود السلحفاة إلى وضعها الابتدائي . وهذا هو الجانب الذي يمكننا التحكم فيه . فعند الوضع الابتدائي تكون السلحفاة متجهة إلى أعلى ويمكن وصف ذلك بأمر لوجو جديد هو HEADING بمعنى اتجاه رأس السلحفاة وهو يقاس بالدرجات . فعند الوضع الابتدائي للسلحفاة

يكون اتجاه الرأس صفراً . وإذا أدخلت الأمر 90 RT يصبح الاتجاه 90 درجة . ويمكنك التأكد من ذلك بتجربة بعض الأوامر مثل :

? RT 90	لإدارة السطح
? PR HEADING	ثم .. طبع اتجاه الرأس
90 ←	الإجابة
? RT 20	وإدارتها مرة أخرى
? PR HEADING	ثم .. طبع الاتجاه الجديد
110	الإجابة

وهكذا

يمكننا الآن إضافة الشرط التالي في البرنامج :

IF HEADING = 0 [STOP]

أنظر الصورة الجديدة للبرنامج .

```

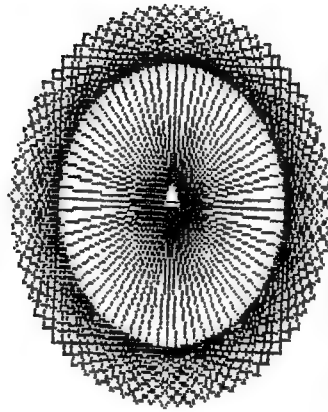
TO SPINSQUARES3 :SIZE :ANGLE
SQUARE :SIZE
RIGHT :ANGLE
IF HEADING = 0 [STOP]
SPINSQUARES3 :SIZE :ANGLE
END

```

الشرط

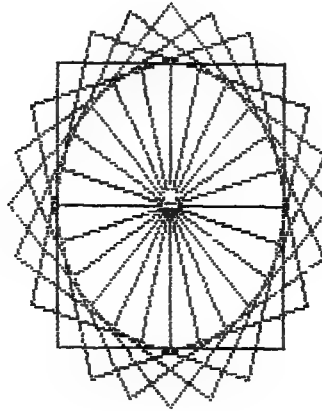
وبذلك يتوقف البرنامج عندما يتم رسم الشكل الكامل مرة واحدة .

وتظهر أهمية هذا الشرط مع بعض الزوايا مثل ٣٥ درجة و ١٥ درجة وهما تعطيان الشكلين أ ، ب الموضحين بعد بالترتيب .



٢٠ درجة

(P1)



١٥ درجة

(B)

● ماذا يحدث بداخل الكمبيوتر؟

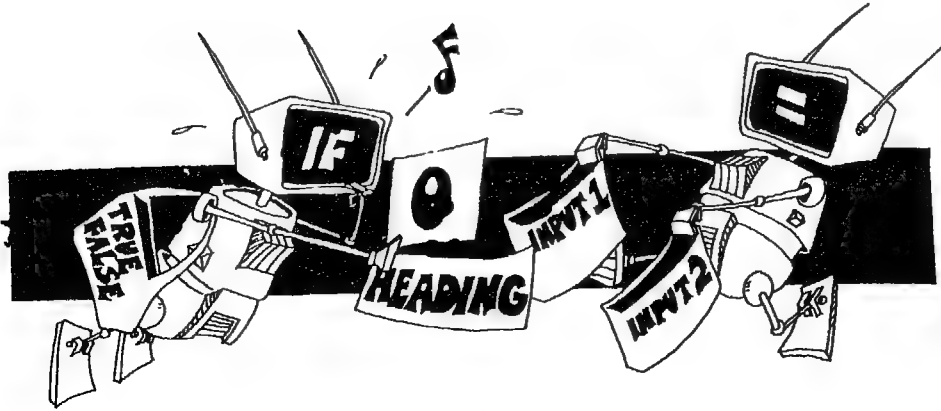
تتبع هذه القصة المصورة الطريقة التي تحكى عن دور العناصر المكونة للعملية الشرطية الآتية :

IF HEADING = 0 [STOP]

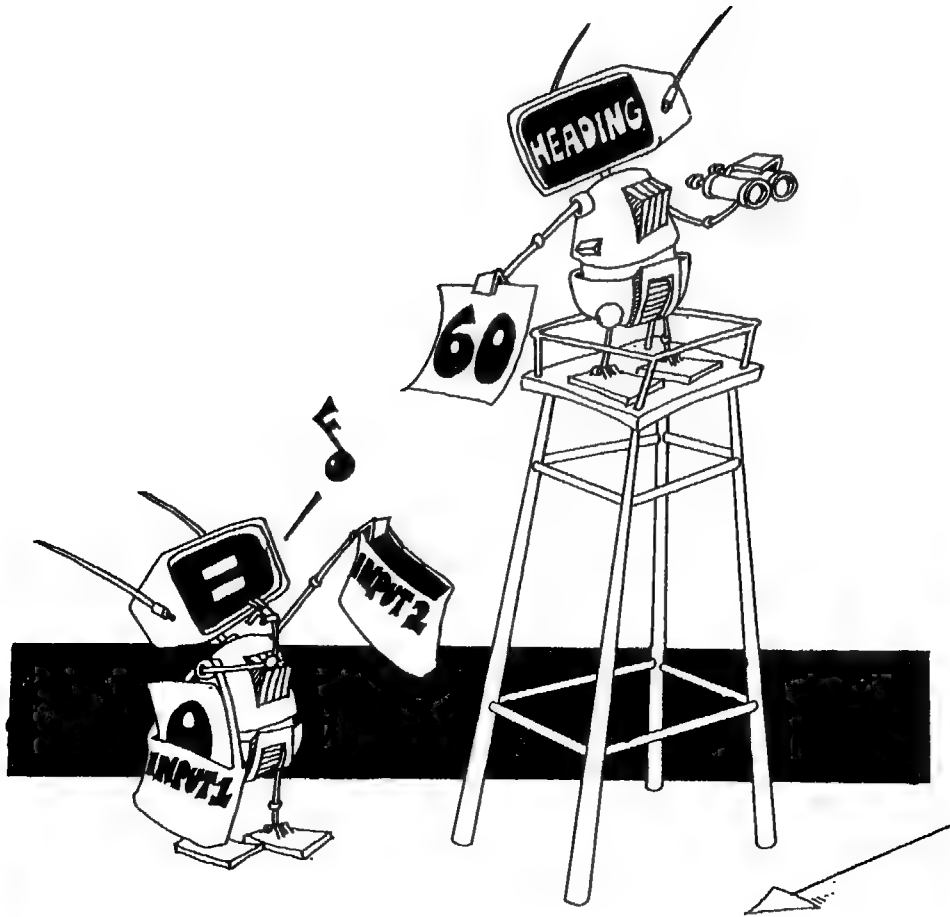
وهذه العناصر هي :

STOP ، = ، HEADING ، IF

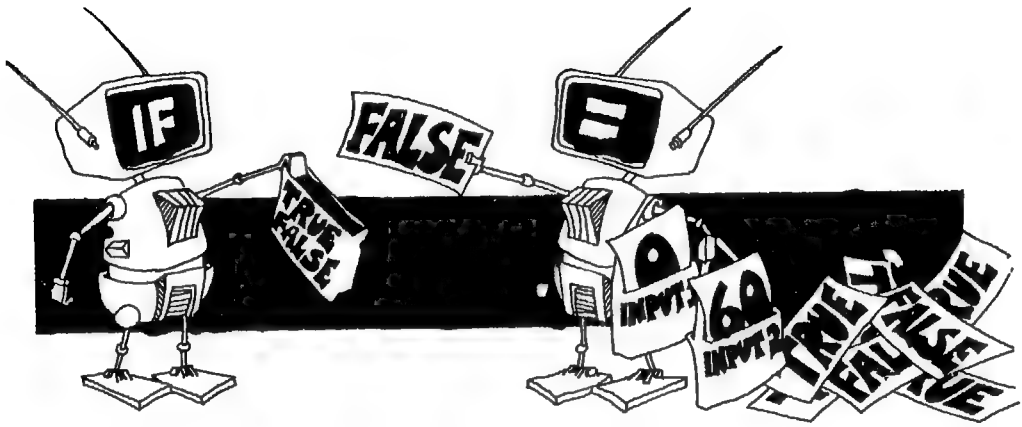
وكلها تسمى أوامر لوجو الأولية (primitives) لذلك نمثلها بالروبوط ذى الشاشة المربعة .



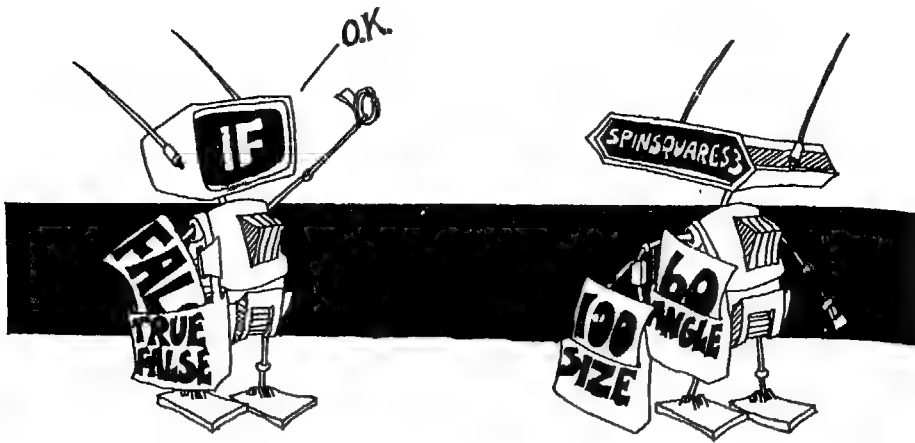
(أ) الروبوط IF يسلم الروبوط (=) مُدخلين هما HEADING ، الصفر . ثم ينتظر منه أحد نتيجتين TRUE أو FALSE .



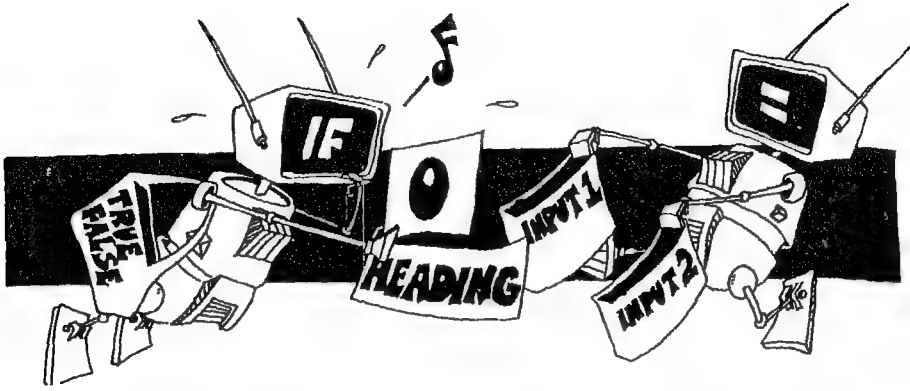
(ب) يقوم الروبوت HEADING بإيجاد قيمة وضع رأس السلحفاه ويسلمه للروبوت (=) ليقارن بينهما .



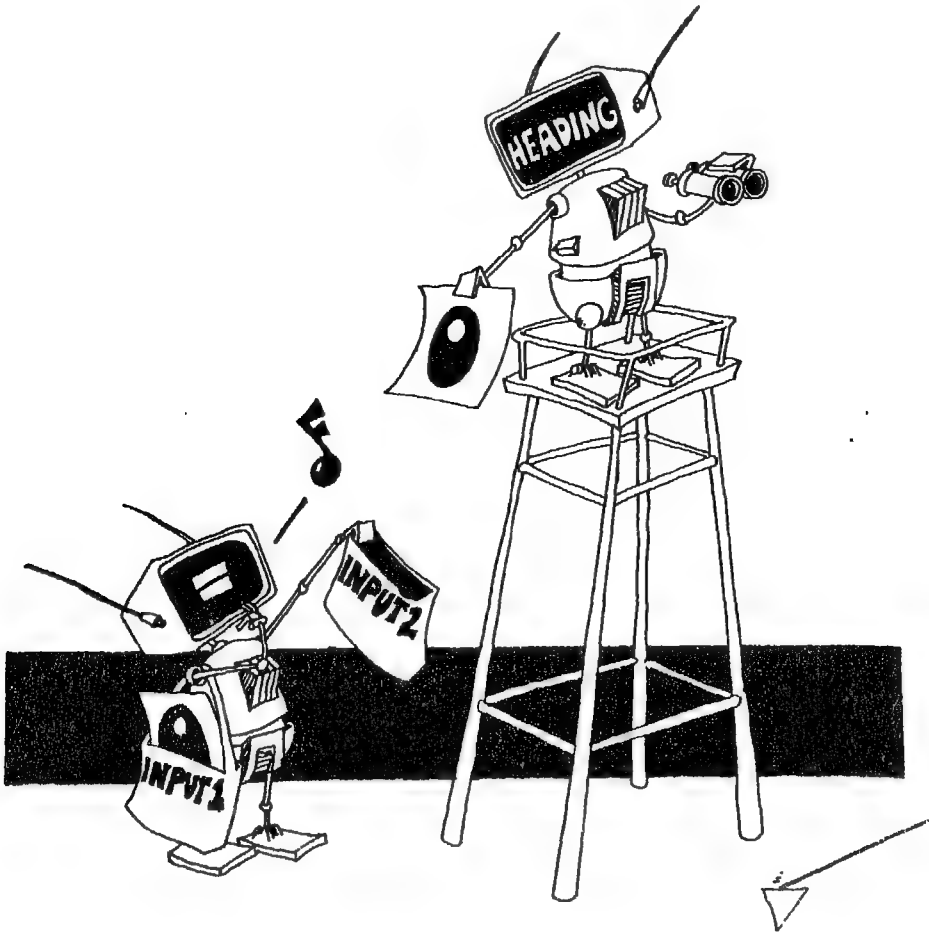
(ج) الروبوت (=) يرسل نتيجة المقارنة إلى الروبوت (IF) وهي غير صحيحة بالطبع FALSE .



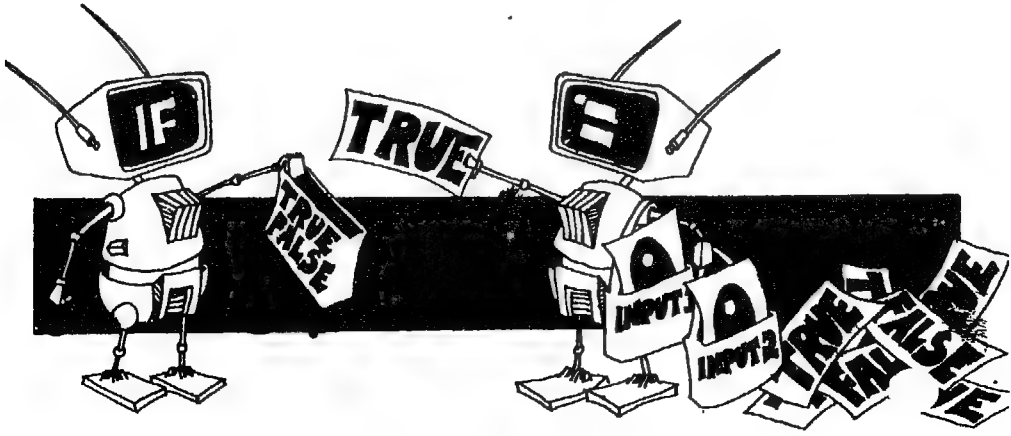
(د) لو كانت نتيجة الاختبار صحيحة TRUE لأمر الروبوت IF بتنفيذ القائمة الأولى [STOP] ولكن نظراً لأن النتيجة هو FALSE فقد أشار الروبوت IF للبرنامج بالاستمرار دون الالتفات إلى القائمة [STOP] ..



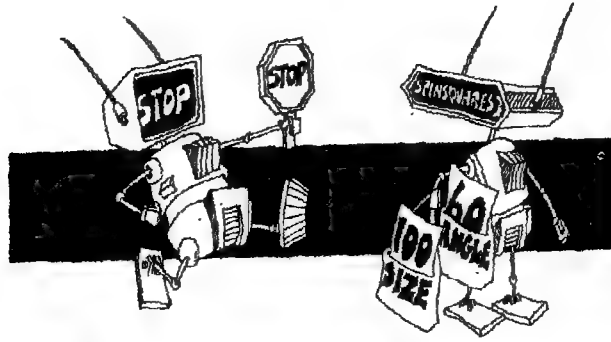
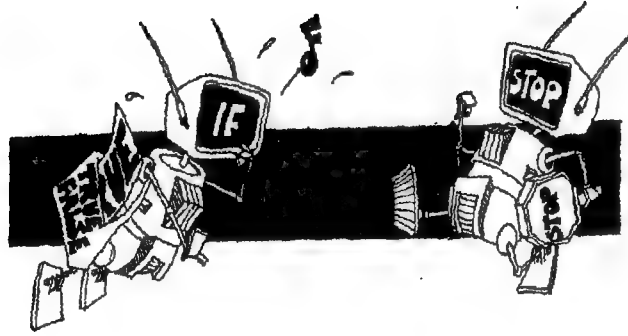
(هـ) تستمر العملية حتى يستلم الروبوت IF المدخلين IF ، 0 مرة أخرى ..



(و) بالبحث والتقصي عن وضع السلاحف هذه المرة وجد الروبوت
HEADING أن رأس السلاحف يشير إلى أعلى فقام بتسليم الرقم صفر للروبوت
(=).



(ز) يقوم الروبوت (=) بإرسال نتيجة الاختبار TRUE إلى الروبوت IF الذي
كلّفه بالمهمة .



(ح) بمجرد أن يتسلم الروبوت IF نتيجة الاختبار TRUE يستدعي فوراً
الروبوت STOP الذي يقوم بدوره بإيقاف البرنامج .

٩

بناء البراج
والألعاب الكومبيوترية

● تغيير وظائف الأزرار

READCHAR (RC)

عند الضغط على أى زر من أزرار الكتابة فإن لبنة واحدة (حرف أو رقم أو علامة خاصة) تُكتب على الشاشة .

ويمكن قراءة اللبنة المدخلة بواسطة العملية READ CHAR وهى تماثل العملية READLIST ولكنها عملية تختص بقراءة لبنة مفردة ولذلك فبمجرد لمس أحد الأزرار تُنفذ العملية ويتم الانتقال إلى الأمر التالى . ويختصر هذا الأمر إلى RC .

وهذه العملية تفتح الباب إمامنا لتطبيقات كثيرة . فالمثال الآتى يستقبل اللبنة من الزر المضغوط ويخصصها للمتغير Z ثم يستخدمها لأداء وظيفة مخالفة تماماً للوظيفة الأصلية . فهو يستخدم الرقم 5 فى إدارة السلحفاه إلى اليسار بمقدار ٩٠ درجة ويستخدم الرقم 6 لتحريك السلحفاه إلى الخلف بمقدار ١٠ خطوات .. وهكذا .. وبذلك نحصل على وسيلة مباشرة للتحكم فى السلحفاه بالأزرار بدلاً من التحكم فيها من خلال برنامج واحد ثابت .

```
TO DRAW
MAKE "Z RC
IF :Z = 5 [ LT 90 ]
IF :Z = 6 [ BK 10 ]
IF :Z = 7 [ FD 10 ]
IF :Z = 8 [ RT 90 ]
DRAW
END
```

● الكلمات والقوائم كمتغيرات :

كما تعاملنا من قبل مع متغيرات الأعداد فإنه يمكن حفظ القوائم والكلمات أيضاً فى متغيرات . والمثال الآتى يوضح إحدى الطرق للتخصيص .

TO SPEAK :MESSAGE
PRINT [THE MESSAGE I AM GOING TO PRINT IS]
PRINT :MESSAGE
END

المتغير

يستقبل هذا البرنامج الرسالة MESSAGE منك ثم يطبعها . والرسالة قد تكون كلمة أو قائمة أو رقما ولكن يراعى طريقة إدخال كل نوعية .
فإذا كنت تعزم إدخال رقم للبرنامج فأنت تكتب اسم البرنامج متبوعاً بالرقم المُدخل كالاتى :

SPEAK 144

أما إذا كنت تنوى إدخال كلمة للمتغير MESSAGE فيم ذلك كالاتى :

SPEAK "HELLO

فإذا كان المُدخل عبارة عن قائمة فإننا نستخدم علامة القائمة كالاتى :

SPEAK [GOOD MORNING SIR]

وهذه الطريقة تماثل تماماً ما اتبعناه مع الأمر MAKE الذى استخدمناه من قبل للتخصيص للمتغيرات سواء مع الأعداد أو الكلمات أو القوائم .
وللتذكرة نقدم المثال الآتى :

MAKE "MESSAGE [THIS IS GETTING SILLY]

المتغير

البيان

ولعلنا نذكر أن المتغير المستخدم مع MAKE لا بد أن يكون كلمة أى أنه يبدأ بالعلامة (") أما البيان فيمكن أن يكون كلمة أو قائمة أو عدداً . بهذا الأمر السابق يتم تخصيص القائمة التى بين القوسين للمتغير MESSAGE .

إن المتغيرات التى تحتوى على كلمات وقوائم مثل المتغيرات التى تحتوى على

أعداد فهي قد تكون متغيرات لبرامج الدوال كما في المثال الأول ، أو متغيرات عامة كما في المثال الثاني .

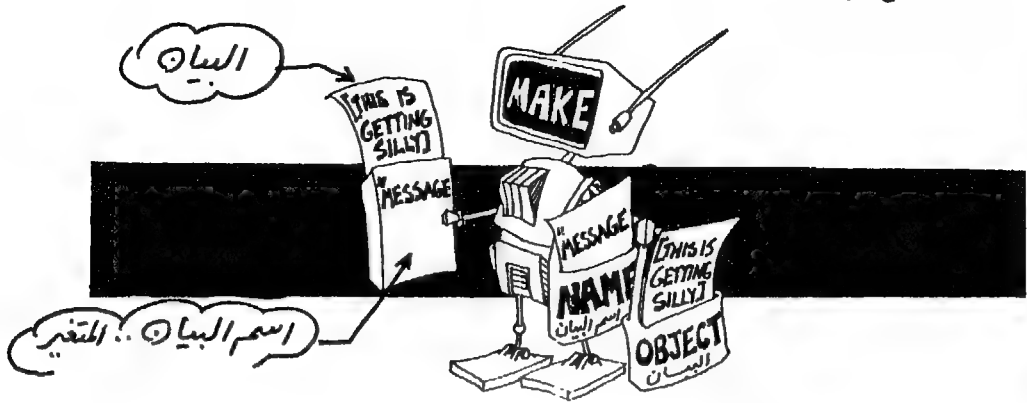
ولعلنا نتذكر طريقتي تخزين كل من المتغيرات العامة ومتغيرات برامج الدوال والفارق بين الطريقتين .

فالميتغيرات العامة تُعتبر مشاعاً في الذاكرة أما متغيرات الدوال فتستخدم داخل برنامجها فقط .

ولعل الساحر لوجو يشرح لنا هذا الأمر بطريقة أفضل .

● ماذا يحدث بداخل الكمبيوتر :

عادة يحمل الروبوت MAKE حقيتين واحدة للبيانات وواحدة لأسماء البيانات (المتغيرات) . لذلك فعند إعطاء الأمر الوارد بالمثال الأخير فإن الروبوت MAKE يقوم بوضع القائمة [THIS IS GETTING SILLY] بداخل الوعاء MESSAGE :



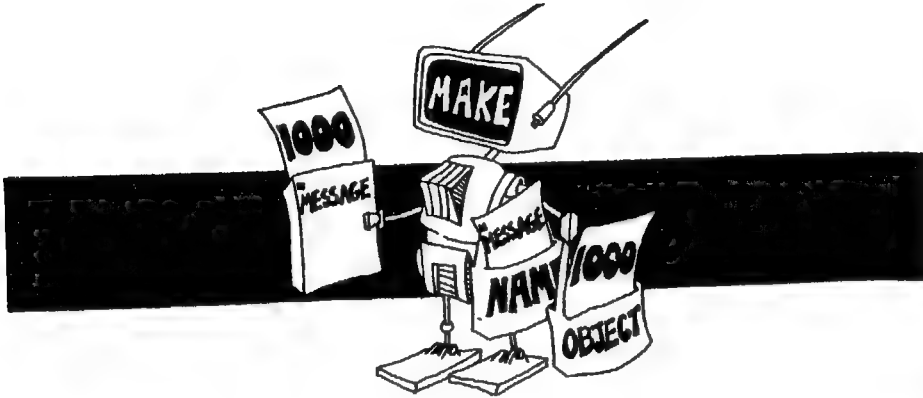
.. ويقوم الساحر لوجو باستلام العبة MESSAGE بما فيها ليخزنها في ذاكرة الكمبيوتر الرئيسية تحت الطلب .

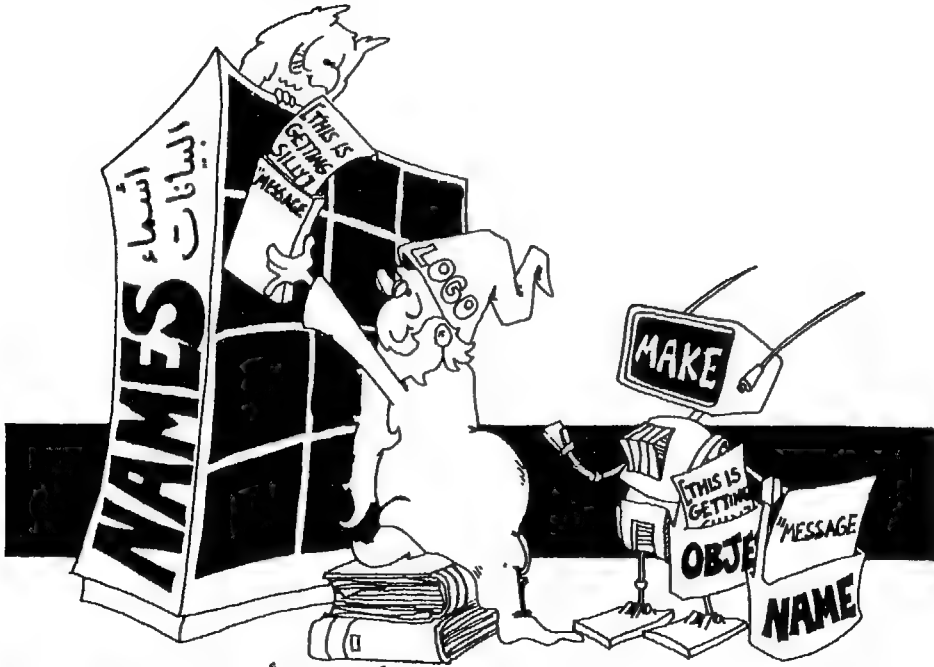


والوعاء MESSAGE يمكن أن نستبدل ما فيه من كلمات بأرقام أيضاً
وبمجرد إعطاء أمر MAKE جديد .

MAKE "MESSAGE 1000

فيقوم على الفور الروبوت MAKE بوضعها في الوعاء MESSAGE تمهيداً
لتسليمه إلى الساحر لوجو .

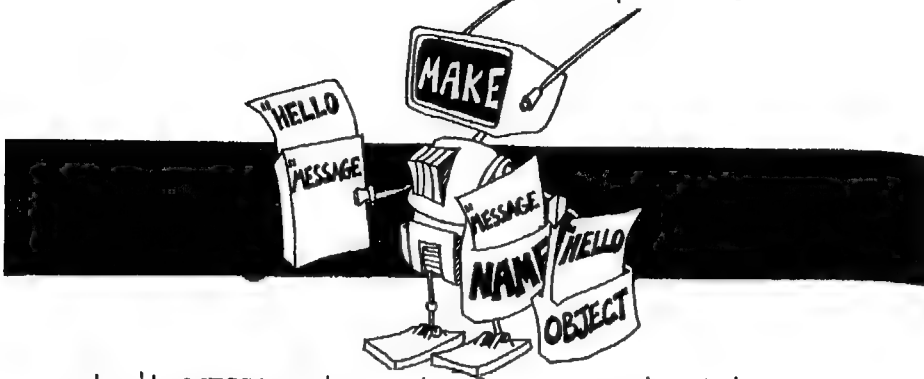




ومحتويات المتغير MESSAGE يمكن تغييرها في أى وقت بأمر MAKE جديد
فبدءاً الروبوت MAKE والساحر لوجو بإعادة العمل مرة أخرى . وها هما
ينفذان الأمر :

MAKE "MESSAGE "HELLO

فها هو الروبوت يتسلم رسالة جديدة .



وها هو الساحر لوجو يستبدل محتويات الوعاء MESS46E بالمحتويات
الجديدة "HELLO" ..



بهذا يصبح وعاء المتغير MESSAGE محتوياً على العدد 1000 . أما البيانات القديمة فذهبت إلى غير رجعة . وللتأكد يمكنك إعطاء الأمر :

PRINT :MESSAGE

سوف نجد الرد جاهزاً وهو 1000 .

بل يمكنك إعطاء هذا الأمر عقب كل تغيير في البيان بالأمر MAKE حتى تشاهد كيف تبدل البيانات في الوعاء MESSAGE .

● جمع الأعداد ووصل القوائم والكلمات :

صادفنا من قبل أمثلة كالآتي :

MAKE "NUMBER 5
PRINT :NUMBER
MAKE "NUMBER :NUMBER + 5
PRINT :NUMBER

أضف 5 إلى قيمة المتغير

فالتغير NUMBER الذى يحتوى على العدد 5 يمكننا أن نضيف إليه (أو نطرح منه ..) ما شئنا من الأرقام دون تغيير اسمه . وقد استخدمنا ذلك من قبل فى إنشاء العدادات .

المتغيرات التى تحتوى على قوائم وكلمات يمكن أيضاً معالجتها بطريقة مماثلة ولكن عندما نتعامل مع البيانات غير العددية فإننا نطلق على عملية الجمع اسماً جديداً هو عملية الوصل (catenation) .

ولغة لوجو تمدنا بوسيلة جاهزة للوصل بين القوائم هى SENTENCE . ولنر معاً هذا البرنامج المشابه للبرنامج السابق .

```
MAKE "MESSAGE [HELLO THERE]
PRINT :MESSAGE
MAKE "MESSAGE SENTENCE :MESSAGE "FRIEND
PRINT :MESSAGE
```

صل المتغير MESSAGE بكلمة FRIEND

واسم المتغير فى حالتنا هذه هو MESSAGE وهو يحتوى على القائمة :

[HELLO THERE]

وفى السطر الثالث من هذا البرنامج نجرى عملية الوصل على كلمة FRIEND المميزة بعلامة الاقتباس المفردة قبلها (") مع قيمة المتغير MESSAGE المميزة بالنقطتين (:). مع استخدام الأمر SENTENCE للوصل بين الكلمة والقائمة .

لذلك عن تشغيل هذا البرنامج نحصل على العبارة الآتية :

HELLO THERE FRIEND

● ماذا يحدث

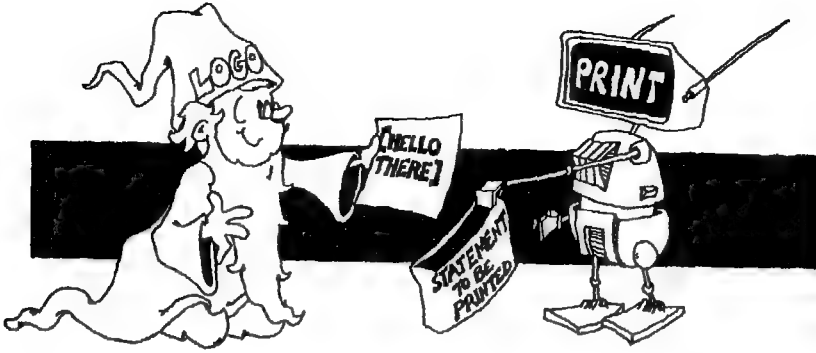
بداخل

الكمبيوتر

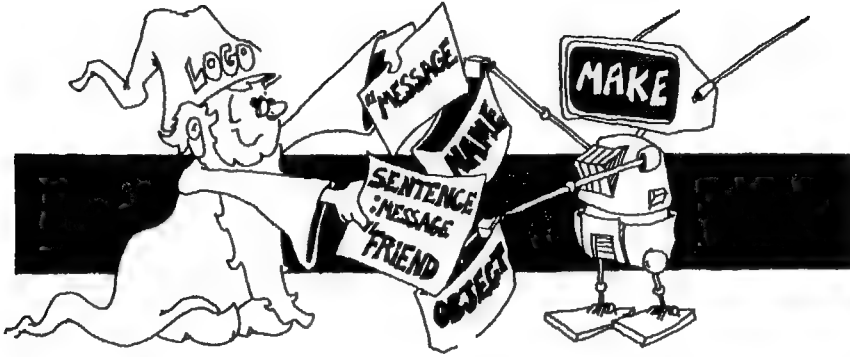
ولنر الآن كيف يقوم الساحر لوجو بهذه العملية :



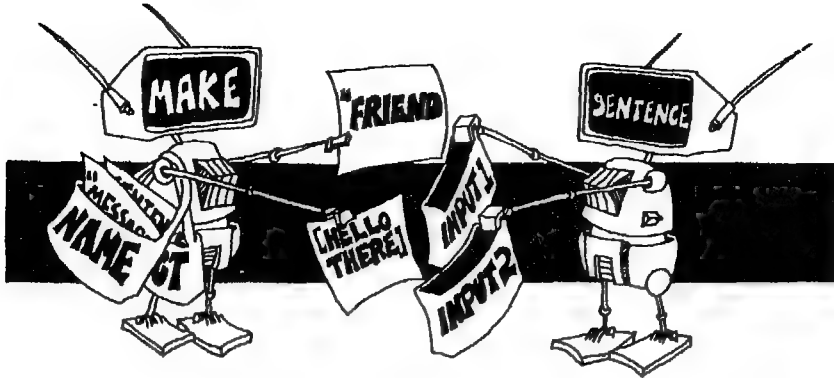
ها هو الساحر لوجو يضع علبة المتغير MESSAGE في الذاكرة .



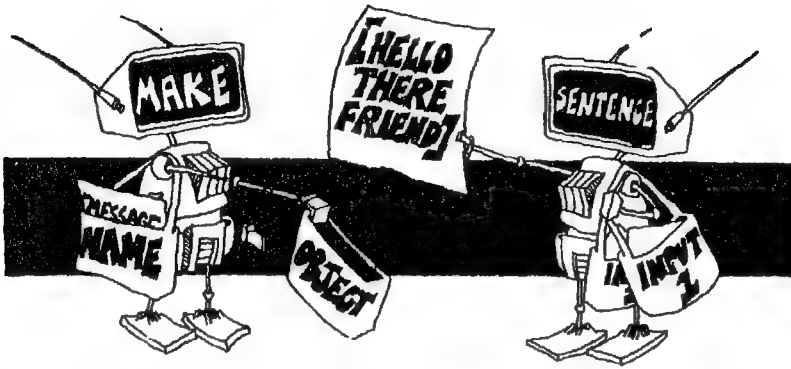
وها هو يلي أمر الطباعة PRINT فيسلم الروبوت PRINT نسخة من القائمة التي نريد طباعتها .



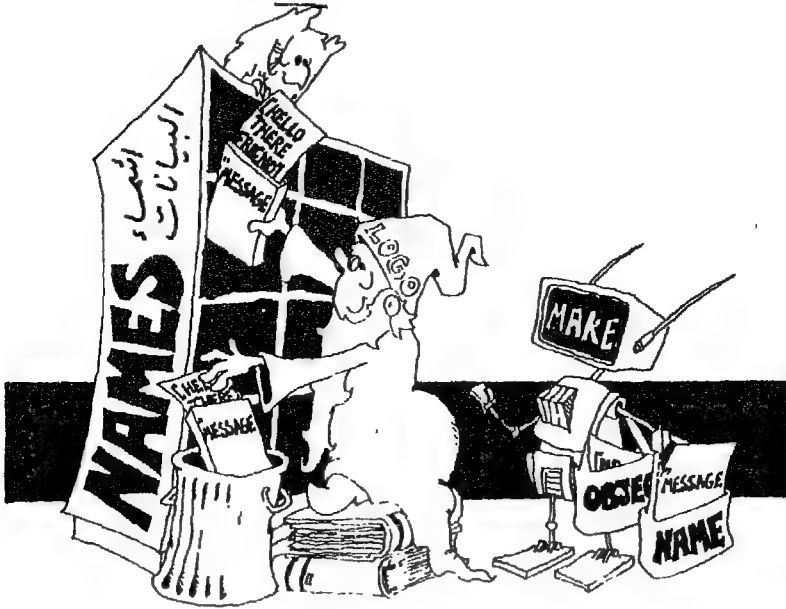
ثم يأتي الروبوت MAKE ومعه عملية جديدة تحتوى على اسم المتغير MESSAGE والبيان المركب "FRIEND :MESSAGE :SENTENCE ولكن البيان لا زال يحتاج إلى مزيد من المعالجة . لذلك يأمر الساحر لوجو الروبوت MAKE باستدعاء الروبوت SENTENCE للقيام بعملية الوصل بين محتوى المتغير MESSAGE وبين الكلمة FRIEND لتكوين الجملة المطلوبة .



يقوم الروبوت MAKE باستدعاء الروبوت SENTENCE ويسلمه بيانات الجملة المطلوب تكوينها في صورة مُدخلين هما "FRIEND و [HELLO THERE] .



يقوم الروبوت SENTENCE بعملية الوصل المطلوبة ويسلم الروبوت
MAKE القائمة [HELLO THERE MY FRIEND] .



وصلت محتويات الجملة SENTENCE الآن .
لذلك لا يحتاج الساحر لوجو للمكونات الأساسية التي كانت موجودة في
علبة المتغير MESSAGE فيلقاها في سلة المهملات .



الآن فقط يستطيع الساحر لوجو أن يستدعي الروبوت PRINT لكي يسلمه
العبارة المطلوب طباعتها كاملة :

HELLO THERE FRIEND

● مثال : برنامج لوصل الكلمات والقوائم :

```
TO GROW :MESSAGE
PRINT [TYPE SOMETHING NEW]
MAKE "NEWPART READLIST
MAKE "MESSAGE SENTENCE :MESSAGE :NEWPART
PRINT [THE MESSAGE IS NOW]
PRINT :MESSAGE
GROW :MESSAGE
END
```

يقوم هذا البرنامج باستقبال ما تدخله إليه من كلمات أو قوائم ووصلها
ببعضها البعض بلا توقف وهذا مثال لاستخدام البرنامج الذي يبدأ بالأمر .

? GROW [HOW]

فيطبع على الشاشة الرسالة (أكتب شيئاً جديداً ..) :

TYPE SOMETHING NEW

وكل ما تكتبه يضمه إلى الكلمة HOW ويطبعه على الشاشة ثم يسألك من
جديد أن تكتب شيئاً جديداً دون توقف .

وفي المثال الموضح نستخدم البرنامج في تكوين العبارة :

HOW ARE YOU GETTING ON

ثم نوقف البرنامج من الخارج .

```
?  
TYPE SOMTHING NEW  
  
?  
?GROW [HOW]  
TYPE SOMTHING NEW  
ARE  
THE MESSAGE IS NOW  
HOW ARE  
TYPE SOMTHING NEW  
YOU  
THE MESSAGE IS NOW  
HOW ARE YOU  
TYPE SOMTHING NEW  
GETTING ON  
THE MESSAGE IS NOW  
HOW ARE YOU GETTING ON  
TYPE SOMTHING NEW
```

STOPPED!!!!. in GROW

إيقاف البرنامج

● استخدام القوائم في ألعاب الكلمات :

العملية الشرطية يمكن أن تستخدم لتنظيم الكثير من العمليات الجارية سواء على الأرقام أو على الكلمات والقوائم .

وأحد أنواع الألعاب الكومبيوترية هي ألعاب الكلمات أو ألعاب المعلومات بدقة أكبر . فالامتحانات بأنواعها تدرج تحت ألعاب المعلومات . ولتر معاً هذا المثال الذي يختبرنا فيما حصلناه من لغة لوجو :

● مثال (١) امتحان لوجو :

TO QUIZ1

- 1 PRINT[WHAT IS THE SHORT FORM OF FORWARD]
- 2 MAKE "ANSWER1 READLIST
- 3 IF :ANSWER=[FD] [PRINT [YOU GOT IT]

STOP]
4 PRINT [NO, NOT CORRECT PLEASE TRY
AGAIN...]
5 QUIZ1
END

وقد رقمنا سطور البرنامج للإيضاح فقط (فيما عدا اسم البرنامج وأمر النهاية) .

١ — ففي السطر الأول يطبع البرنامج على الشاشة العبارة التي تحتويها القائمة وهي :

WHAT IS THE SHORT FORM OF FORWARD

بمعنى ما هو اختصار الأمر FORWARD ?

٢ — ويتلقى الكمبيوتر الإجابة في السطر الثاني بموجب الأمر READLIST ويخصصها للمتغير ANSWER1 بعملية التخصيص MAKE .

٣ — يقوم البرنامج في السطر الثالث باختبار محتويات المتغير ANSWER1 بواسطة العملية الشرطية .

فإذا كانت محتويات المتغير ANSWER1 مساوية للقائمة FD كانت الإجابة صحيحة وقام الكمبيوتر بتنفيذ القائمة المركبة التالية وهي تتكون من جزئين أولهما طبع الرسالة « YOU GOT IT » والجزء الثاني هو الأمر STOP الذي يجعله يتوقف .

٤ — أما إذا لم تكن الإجابة هي FD فسوف ينفذ السطر الرابع الذي يحتوى على الرسالة « الإجابة غير صحيحة حاول مرة أخرى .. » .

٥ — وبذلك يتم الانتقال إلى أول البرنامج بموجب الأمر QUIZ1 في السطر الخامس الذي هو عبارة عن اسم البرنامج نفسه .

● مثال (٢) امتحان لوجو C :

أما هذا البرنامج فهو صورة مطوّرة للبرنامج السابق حيث يمنحك اختياراً أن تحاول مرة أخرى أو لا تحاول .

TO QUIZ2

- 1 PRINT [WHAT IS THE SHORT FORM OF FORWARD?]
- 2 MAKE "ANSWER2 READLIST
- 3 IF :ANSWER2 = [FD] [PRINT [CORRECT!] STOP]
- 4 PRINT [NO, THAT'S NOT IT.]
- 5 PRINT [WOULD YOU LIKE TO TRY AGAIN?]
- 6 MAKE "TRY READLIST
- 7 IF :TRY = [NO] [PRINT [THE ANSWER IS: FD] STOP]
- 8 IF :TRY = [YES] [QUIZ2 STOP]
- 9 PRINT [I QUIT! YOU DIDN'T ANSWER YES OR NO.]
- END

عند تشغيل هذا البرنامج فإنه يطبع الرسالة التى تحتوى عليها القائمة فى السطر الأول :

WHAT IS THE SHORT FORM OF FORWARD?

بمعنى ما هى الصيغة المختصرة للأمر FORWARD ؟

فى السطر الثانى يتلقى منك الكمبيوتر الإجابة بالأمر READLIST ويخصصها للمتغير ANSWER2 .

بعد ذلك يتم اختيار محتوى المتغير ANSWER2 فى السطر الثالث الذى يحتوى على أوامر العملية الشرطية . فإذا كانت الإجابة هى FD فإن الكمبيوتر يطبع الرسالة CORRECT ويتوقف بموجب الأمر STOP . وإلا ..

وإلا فإنه يطبع الرسالة الواردة فى السطر الرابع :

NO THAT'S NOT IT

وبلى ذلك طبع السؤال الذى فى السطر الخامس « هل تحاول مرة أخرى ؟ » .

WOULD YOU LIKE TO TRY AGAIN ?

ويستقبل إجابة جديدة في السطر السادس ويضعها في المتغير TRY
وفي السطر السابع يختبر محتوى المتغير TRY بالعملية الشرطية . فإذا كان
يحتوى على القائمة [NO] فإن الكمبيوتر يطبع الإجابة الصحيحة ثم يتوقف .
وفي السطر الثامن يختبر ما إذا كانت الإجابة هي [YES] . فإذا كانت
كذلك يعود إلى أول البرنامج لتنفيذ الأمر الأول في القائمة وهو :

QUIZ2

وهذا هو اسم اللعبة نفسها .

أما السطر التاسع فيتم تنفيذه في حالة ما إذا كانت الإجابة شيئاً مختلفاً غير
(YES) أو (NO) فهذان هما الإجابتان اللتين يقبلهما الكمبيوتر أما ما عدا ذلك
فإنه يفضض ويرسل الرسالة التي يحتوى عليها السطر التاسع وهي تقول
« سوف أرحل لأنك لم تجبني بنعم أولاً . وينتهي البرنامج عند هذا الحد .

OR, AND

● العمليات المنطقية

إذا أردت السفر من القاهرة إلى الإسكندرية فيمكنك استخدام القطار أو
« الأوتوبيس » أينما تجد مقعداً خالياً .

فإذا أردت إرسال إنسان آلى (روبوت) ليقوم بشراء تذكرة سفر لك فعليك
أن تشرح له هذه المسألة بلغة يفهمها .

وفي لغة لوجو نستخدم الأمر OR الذى يدخل مع الأمر IF في عملية مركبة
للتعبير عن ذلك :

IF OR مقعد في القطار مقعد في الأتوبيس احجز
الشرط الأول الشرط الثانى التصرف فى أى من الحالىن

والشرط الأول هنا هو وجود مقعد خال بالقطار .

والشرط الثانى هو وجود مقعد خال بالأتوبيس .

والتصرف المطلوب هو الحجز .

فلو تحقق أحد الشرطين أو كلاهما فسوف يتم الحجز والسفر .

وتسمى هذه العملية بعملية الجمع المنطقى لأنها . من وجهة نظر الكمبيوتر عملية منطقية لها أحد نتيجتين إما صحيح TRUE أو خطأ FALSE فهو يختبر الشرط الأول فإذا وجده صحيحاً أى كانت نتيجته TRUE اكتفى به واعتبر أن العملية كلها صحيحة ولذلك فهو ينفذ التصرف المطلوب .


فإذا لم يتحقق الشرط الأول (أى كانت نتيجته FALSE) اختبر الشرط الثانى أيضاً فإذا كان صحيحاً قام أيضاً بتنفيذ التصرف المطلوب . حالة واحدة هى التى تفشل فيها العملية OR وهى عندما لا يجد الروبوت مقعداً خالياً فى القطار أو فى الأتوبيس . بذلك يعطى كل من الشرطين النتيجة FALSE ولا تتحقق العملية الشرطية فينفذ القائمة التالية بعد ذلك ، أياً كان مضمونها .

هناك عملية منطقية أخرى هى العملية AND وهى أكثر تشدداً من العملية السابقة .

ولنفرض أنك طلبت من الروبوت تحقيق شرط آخر وهو أى موعد القطار قبل الرابعة مساء .

بهذا تخضع عملية حجز تذكرة القطار لشرط مركب وهو وجود مقعد خال وأن يكون الموعد قبل الرابعة .

هذا يتم التعبير عنه على الصورة :

	الحجز	قبل الرابعة مساء	مقعد فى القطار
IF AND			
	التصرف المطلوب	الشرط الثانى	الشرط الأول
	فى حالة تحقق الشرط		

وتسمى العملية AND بعملية الضرب المنطقى .

فلنر مثلاً بسيطاً باستخدام الأعداد :

```
IF OR 1=1 2=5 [ PRINT [ OK, SIR ] STOP ]  
PRINT [ NOT OK ]
```

ماذا سيطبع الكمبيوتر في هذا المثال الرسالة الأولى أم الثانية ؟

إن الشرط الأول $1 = 1$ يعطى القيمة TRUE .

أما الشرط الثانى $2 = 5$ فيعطى القيمة FALSE .

ونظراً لأن الأمر OR يكتفى بتحقيق شرط واحد لذلك فهو سيعتبر أن العملية الشرطية قد نجحت ويعطى الأمر OR القيمة TRUE كنتيجة نهائية فيطبع بذلك الرسالة الأولى، وهى :

OK , SIR

ثم يتوقف البرنامج .

ولو استبدلنا الشرط $1 = 1$ بشرط آخر مثل $1 = 2$ فإن الرسالة الثانية هى التى ستطبع .

ولنر مثلاً للعملية AND باستخدام الأرقام أيضاً .

```
IF AND 1 = 1 2 = 5 [ PRINT [ OK, SIR ] STOP ]  
PRINT [ NOT OK ]
```

في هذه الحالة لن تتحقق نتيجة العملية الشرطية لأن الأمر AND سوف يُرجع القيمة FALSE لأن أحد الشرطين لم يتحقق .

● مثال (٣) امتحان لوجو ٣ :

وبذلك نصل إلى تطوير جديد في هذا المثال حيث أن السؤال الواحد في بعض الأحيان قد تكون له أكثر من إجابة صحيحة . ولنر هذا السؤال :

” كم من المُدخلات (INPUTS) يحتاج إليها الأمر MAKE ؟ “ والإجابة على هذا السؤال قد تكون بالحروف (TWO) أو بالأرقام (2) وكلاهما إجابة صحيحة .

ولتلقين الكمبيوتر بهذه الاحتمالات يجب استخدام الأمر (OR) .
ولنر المثال :

TO QUIZ3

- 1 PRINT [HOW MANY INPUTS DOES THE]
 - 2 PRINT [MAKE COMMAND NEED?]
 - 3 MAKE "ANS3 READLIST
 - 4 IF OR :ANS3 = [TWO] :ANS3 = [2] [PRINT [RIGHT.] STOP]
 - 5 PRINT [NO, MAKE NEEDS TWO INPUTS,]
 - 6 PRINT [A NAME AND A VALUE]
- END

١ ، ٢ — الرسالة التي تظهر على الشاشة في البداية قد تم طبعها في السطرين 1 ، 2 وسوف تظهر مكتوبة على سطرين كما هي .

٣ — تُخصص الإجابة للمتغير ANS3 .

٤ — يتم اختبار الإجابة ANS3 ما إذا كانت مساوية [2] أو [TOW] وفي أى من الحالتين تطبع الرسالة RIGHT ثم يتوقف البرنامج .

٥ ، ٦ — في حالة احتواء الإجابة ANS3 على أى شيء خلاف القيمتين [2] أو [TWO] تطبع الرسالة :

NO, MAKE NEEDS TWO INPUTS

A NAME AND A VALUE

ويشهى البرنامج عند هذا الحد .

مثل هذا البرنامج الذى يضع الاحتمالات المختلفة للإجابات يوصف بأنه برنامج ذكى . أما البرنامج الذى يعترف بالرقم 2 ولا يعترف بكلمة TWO فهو برنامج غبى

وبالطبع لا يتوقف الأمر على مجرد احتمالين فهناك بعض الأسئلة تختمل إجابات كثيرة أو ربما تحتاج إلى مناقشة وفي هذه الحالة يجب تغذية الكمبيوتر بكمية كبيرة من المعلومات .

ومن خصائص الامتحان الذكي أيضاً تتبع للإجابات جميعاً وتقييمها لإعطاء درجة نهائية أو تقدير نهائي .

وبوضع العديد من البرامج الفرعية التي يحتوى كل منها على سؤال مستقل يمكن بناء امتحان متكامل ببرنامج رئيسي بالصورة الآتية :



TO LOGOQUIZ
QUIZ1
QUIZ2
QUIZ3
QUIZ4
QUIZ5
...
...
...
END

● تمرير البيانات من برنامج إلى آخر OUTPUT (OP)

لعلنا لم نلتق من قبل بلفظة « التمرير » إلا في مباريات كرة القدم حيث يقوم أحد اللاعبين بتمرير الكرة إلى لاعب آخر ليضعها في المرمى مسجلاً هدفاً .

ونحن الآن أمام متغير وظيفته تنحصر أساساً في « تمرير » قيم المتغيرات إلى البرنامج الرئيسي لكي يستخدمها المبرمج فيما يشاء .

ولنر المثال الآتي :

```

TO MEAN :X :Y
OUTPUT (:X + :Y)/2
END

```

هذا البرنامج لا يطبع شيئاً على الشاشة ولكنه يحسب قيمة التعبير الرياضي ويمرر قيمته إلى البرنامج (أو الأمر المباشر) الذي يستدعيه كالمثال الآتي :

```

PR MEAN 6 3 ← الأمر
4.5 ← الجواب

```

وكما نرى فإن عملية التحرير تتم عندما يُستدعى البرنامج المحتوى على الأمر OUTPUT .

لذلك نرى أن هناك فارقاً بين البرنامج الذي ينتهى بالأمر END والبرنامج الذي ينتهى بالأمر OUTPUT . فالأول يُعامل كأنه أمر من أوامر لوجو ويتم تنفيذه بمجرد كتابة اسمه . أما الأخير فإنه يعتبر عملية مثل عملية الجذر التربيعي وجب الزاوية ولا يستخدم إلا من خلال برنامج أو أمر مباشر .

والأمر OUTPUT يعتبر نهاية البرنامج لأنه يوقف التنفيذ تماماً كما يفعل الأمر STOP مع فارق أنه يضع في الذاكرة قيمة المتغير أو التعبير الذي يتبعه والذي نطق عليه اسم « الهدف » (object) .

● اختبار الأعداد NUMBERP

أحياناً نحتاج لمعرفة ما إذا كان أحد المتغيرات يحتوى على قيمة عددية أم على قائمة . فمثلاً عندما نقرأ عدداً ما بالأمر READLIST فإنه يُخزن في صورة قائمة ولا يمكن إجراء أية عمليات حسابية عليه في هذه الحالة .

ويمكن إجراء هذا الاختبار بالأمر NUMBERP كالاتي :

```

PR NUMBERP 3 ← الأمر
TRUE ← الجواب

```

● مثال ٢ : الأمر ← **PR NUMBERP [3]**

FALSE ← الجواب

● مثال ٣ : **TO TEST1**

MAKE "N 55

PR NUMBERP N

END

عند تشغيل هذا البرنامج يعطى القيمة TRUE

● مثال ٤ : **TO TEST2**

MAKE "N RL

PR NUMBERP N

END

وعند تشغيل هذا البرنامج فإنه يعطى القيمة FALSE

● النفي المنطقي NOT :

يستخدم النفي المنطقي لعكس النتيجة كالأمثلة الآتية :

PRINT 4 = 5 ← الأمر

FALSE ← الجواب

PRINT NOT 4 = 5 ← الأمر

TRUE ← الجواب

PRINT NOT NUMBERP 3 ← الأمر

FALSE ← الجواب

المثال الأخير معناه « هل البيان 3 لا يُعتبر عدداً ؟ » والإجابة بالطبع هي « بلى » .

GUESSNUMBER

● لعبة الرقم السري

في عقل الكمبيوتر رقم معين يعرفه لكننا لا نعرفه ومطلوب منا أن نخمن هذا الرقم والكمبيوتر يتلقى إجابتنا مرة بعد مرة ويحاول جاهداً أن يقربنا من الإجابة الصحيحة .

لنر أولاً كيف يعمل هذا البرنامج على الشاشة :

I AM THINKING OF A NUMBER BETWEEN 0 AND 100. SEE IF YOU CAN GUESS IT.

> 50

TOO LOW

> 75

TOO LOW

> 85

TOO HIGH

> 80

TOO LOW

> 82

TOO HIGH

> 81

GOT IT!

محاولة الأولى

الكمبيوتر: الرقم صغير جداً

المحاولة الثانية

الكمبيوتر: الرقم كبير جداً

الكمبيوتر: لقد نجحت في التخمين

رسالة البداية على الشاشة

يبدأ الكمبيوتر بطبع الرسالة الموضحة على الشاشة يقول فيها :

« إنني أفكر في رقم بين الصفر والمائة ... هل تستطيع تخمينه ؟ » . ثم تحاول أنت أن تكتب رقماً جازافياً مثل 50 .

فيرد عليك الكمبيوتر قائلاً : « هذا الرقم أصغر بكثير » .

وهذا يجعلك في المحاولة التالية تكتب رقماً أكبر .. مثلاً 75 .

فيقول .. « هذا الرقم أصغر بكثير » .

إذن نحاول أن نكتب رقماً أعلى من ذلك .. مثلاً 85 .

فيرد علينا قائلاً « هذا الرقم كبير جداً » .

وهكذا حتى تتحقق المحاولة الناجحة .. وفي هذا المثال تكون 81 .

● فيكيف ننشئ هذا البرنامج ؟ إن العمل سوف يتكون من أجزاء أربعة هي :

١ — كتابة تعليمات اللعبة على الشاشة .

٢ — اختيار رقم ما بين 0 ، 100 .

٣ — استقبال تخمينات مستخدم البرنامج .

٤ — إجراء اختبار على كل رقم يدخله المستخدم وطبع رسالة تخبره ما إذا كان الرقم أكبر أو أصغر من الرقم المقصود . وعلى ذلك يمكننا بناء البرنامج كالآتي :

TO GUESSNUMBER
INSTRUCTIONS
CHOOSENUMBER
GETGUESS
END

البرنامج الرئيسي

1 TO INSTRUCTIONS
CLEARTEXT
PRINT [I AM THINKING OF A NUMBER BETWEEN 0]
PRINT [AND 100. SEE IF YOU CAN GUESS IT.]
END

2 TO CHOOSENUMBER
MAKE "NUMBER 1 + RANDOM 99
END

3 TO GETGUESS
TYPE ">
MAKE "GUESS READNUMBER
CHECKGUESS :GUESS :NUMBER
END

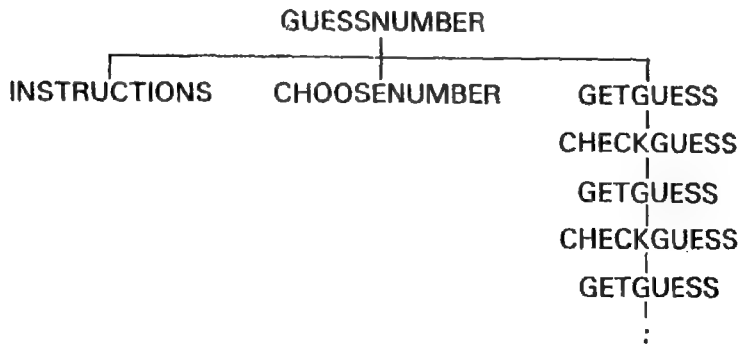
4 TO CHECKGUESS :GUESS :NUMBER
IF :GUESS = :NUMBER [PRINT [GOT IT] STOP]
IF :GUESS > :NUMBER [PRINT [TOO HIGH] GETGUESS STOP]
IF :GUESS < :NUMBER [PRINT [TOO LOW] GETGUESS STOP]
END

البرامج الفرعية

البرنامج الرئيسى هو GUESSNUMBER

وهو يستدعى ٣ برامج فرعية : INSTRUCTIONS للتعليمات .
CHOOSENUMBER لاختيار الرقم GETGUESS لاختبار الرقم المُدخل .

والبرنامج الأخير يستدعى برنامجاً فرعياً آخر يحتوى على العمليات الشرطية المختلفة ويتناوب البرنامجان الأخيران استدعاء بعضهما البعض عدة مرات وحتى الوصول للرقم المطلوب . أى أنه يمكن تصور الشجرة الآتية لتنفيذ البرامج :



● ولنناقش الآن البرامج الفرعية .

تظهر لنا مبدئياً عند القراءة الأولى لهذه البرامج الفرعية عبارات جديدة من لغة لوجو . هى : READNUMBER , CLEARTEXT , TYPE .

*** الأمر TYPE**

هذا الأمر يماثل تماماً الأمر PRINT ولكنه لا يجعل النقطة الضوئية (cursor) تنتقل إلى السطر التالى بمعنى أن الطباعة المتتالية بالأمر TYPE تكون على نفس السطر .

*** الأمر CLEARTEXT واختصاره CT**

هذا الأمر يستخدم لتنظيف الشاشة من الكتابة عندما تكون الشاشة فى طور الكتابة (TEXTSCREEN) .

* الأمر READNUMBER

يكافئ الأمر READLIST ولكنه مخصص لقراءة الأرقام فهو يجعل الكمبيوتر ينتظر إدخال رقم من الأرقام علاوة على أنه يقوم باختبار ما إذا كان المدخل هو رقم فعلاً أم قائمة أو حتى قائمة خالية [] .

وهذا الأمر ليس من الأوامر القياسية للغة لوجو . بل هو برنامج فرعى ضمن البرامج التي يحتوى عليها القرص (LWIL) . وقد عرضنا نص هذا البرنامج في الملحق رقم (٣) ويمكنك نسخه واستخدامه كما هو .

ولكنه قد يتصادف أيضاً أن يحتوى البرنامج الفرعى READNUMBER على أوامر لا تتوفر في طراز اللغة الذى يستخدمه كومبيوتر معين مثل الأمر TEST وفى هذه الحالة يفضل بناء البرنامج بنفسك باستخدام الأوامر المتاحة بطراز اللغة الذى تعمل عليه .

والبرنامج الآتى بعد يحمل نفس الاسم READNUMBER أيضاً ولكنه يستخدم العملية الشرطية IF بدلاً من الأمر TEST حتى يصلح لأجهزة الكمبيوتر الصغيرة مثل سنكلير . ولا بأس من استخدام هذا البرنامج الفرعى أيضاً .

```
TO READNUMBER
MAKE "NUM1 RL
IF :NUM1 = [] [PR [PLEASE TYPE A
NUMBER] OUTPUT READNUMBER]
IF NOT NUMBERP FIRST :NUM1 [PR
PLEASE TYPE A NUMBER] OUTPUT REA
DNUMBER]
OUTPUT FIRST :NUM1
END
```

البرنامج الفرعى READNUMBER

هذه هي الأوامر الجديدة التى نراها فى مجموعة البرامج الفرعية للعبة للرقم السرى .

أما اللعبة نفسها فتسير أحداثها كالآتي :

اللعبة تتكون أساساً من برنامج رئيسي هو GUESSNUMBER الذي يستخدم ٣ برامج فرعية كالآتي :

```
TO GUESSNUMBER
INSTRUCTIONS
CHOOSENUMBER
GETGUESS
END
```

● البرنامج الفرعي الأول هو برنامج التعليمات INSTRUCTIONS الذي يقوم بطباعة تعليمات اللعبة على الشاشة :

```
TO INSTRUCTIONS
CLEARTEXT
PRINT [I AM THINKING OF A NUMBER BETWEEN 0]
PRINT [AND 100. SEE IF YOU CAN GUESS IT.]
END
```

في هذا البرنامج نستعمل بمسح الشاشة في طور الكتابة بالأمر CLEARTEXT
تم نطبع تعليمات اللعبة على سطرين متعاقبين .

● يلي ذلك البرنامج الفرعي لاختيار الرقم السري CHOOSENUMBER

```
TO CHOOSENUMBER
MAKE "NUMBER 1 + RANDOM 99
END
```

وفي هذا البرنامج نستخدم الأمر RANDOM لتوليد عدد يتراوح بين الصفر و 99. ثم نضيف إليه الرقم 1 فتصبح قيمته عندئذ بين الصفر و 99 . ونخصص هذا العدد العشوائي للمتغير NUMBER بواسطة أمر التخصيص MAKE .

● البرنامج الفرعي التالي هو GETGUESS الذي يستقبل الرقم المدخل من مستخدم البرنامج ويجري عليه الاختبارات اللازمة :


```

      TO GETGUESS
1  TYPE ">"
2  MAKE "GUESS READNUMBER
3  CHECKGUESS :GUESS :NUMBER
      END

```

يبدأ البرنامج بطبع العلامة (>) : الأمر TYPE وذلك حتى لا تتفعل النقطة الضوئية إلى السطر التالي . وبذلك فإن أى كتابة تالية سوف تكون على نفسه السطر .

أما السطر التالي من البرنامج فهو يخصص الرقم المُدخل للمتغير GUESS . وفى السطر الثالث يتم استدعاء البرنامج الفرعى GHECKGUESS الذى يستخدم دليلين أحدهما هو المتغير GUESS الذى يحتوى على الرقم المُدخل بواسطة المستخدم والثانى هو MUMBER وهو الرقم السرى الذى يخبئه الكمبيوتر . ومن المتوقع أن تتم المقارنة بين الرقمين بواسطة البرنامج الفرعى الأخير .

● البرنامج الفرعى CHECKGUESS لاختبار صحة التخمين :

```

      TO CHECKGUESS :GUESS :NUMBER
1  IF :GUESS = :NUMBER [PRINT [GOT IT] STOP]
2  IF :GUESS > :NUMBER [PRINT [TOO HIGH] GETGUESS STOP]
3  IF :GUESS < :NUMBER [PRINT [TOO LOW] GETGUESS STOP]
      END

```

كما نرى أن هذا البرنامج هو برنامج لدالة تستخدم دليلين GUESS ، NUMBER وتقرن بينهما باستخدام العملية الشرطية .

وفى السطر الأول يتم اختبار التساوى بين الرقم المُدخل والرقم السرى فإذا صحّ الاختبار فإن الرسالة "GOT IT" تظهر على الشاشة وينتهى الأمر .

أما إذا كان الرقم أكبر كما فى السطر الثانى فإن الرسالة "TOO HIGH" هى التى تطبع ويتم التفرع إلى البرنامج GETGUESS لاستقبال رقم جديد .

وأما إذا كان الرقم أصغر فتطبع الرسالة "TOO LOW" ويتم التفرع أيضاً

إلى البرنامج GETGUESS لاستقبال رقم جديد .

وعند العودة إلى البرنامج GETGUESS يستقبل البرنامج تخميناً جديداً ثم يعود لاختباره بواسطة البرنامج CHECKGUESS وهكذا حتى نصل إلى الرقم الذى فى عقل الكمبيوتر .

والآتى بعد هو طبعة لكل ما فى حيز العمل أثناء تشغيل هذه البرامج ونلاحظ ضرورة وجود البرنامج READNUMBER فى حيز العمل . كما نرى مجموعة المتغيرات العامة التى استخدمت فى البرامج علاوة على بعض متغيرات أخرى مثل L ، S وهى تابعة لبرامج أخرى تم مسحها من بالأمر (ER) .

```

?
TO GUESSNUMBER
TO CHECKGUESS :GUESS :NUMBER
TO INSTRUCTIONS
TO GETGUESS
TO CHOOSENUMBER
TO READNUMBER

?
TO GUESSNUMBER
INSTRUCTIONS
CHOOSENUMBER
GETGUESS
END

TO CHECKGUESS :GUESS :NUMBER
IF :GUESS = :NUMBER (PRINT (GOT
IT) STOP)
IF :GUESS > :NUMBER (PRINT (TOO
HIGH) GETGUESS STOP)
IF :GUESS < :NUMBER (PRINT (TOO
LOW) GETGUESS STOP)
END

TO INSTRUCTIONS
CT
PR (I AM THINKING OF A NUMBER BE
TWEEN 0)
PR (AND 100 . SEE IF YOU CAN GUE
SS IT)
END

TO GETGUESS
TYPE ">
MAKE "GUESS READNUMBER
CHECKGUESS :GUESS :NUMBER
END

TO CHOOSENUMBER
MAKE "NUMBER 1 + RANDOM 99
END

TO READNUMBER
MAKE "NUM1 RL
IF :NUM1 = [] (PR (PLEASE TYPE A
NUMBER) OUTPUT READNUMBER)
IF NOT NUMBERP FIRST :NUM1 (PR (
PLEASE TYPE A NUMBER) OUTPUT REA
DNUMBER)
OUTPUT FIRST :NUM1
END

MAKE "GUESS "70
MAKE "L "555
MAKE "S "14.323957
MAKE "NUMBER "6
MAKE "NUM1 [6]

```



تجربة



هل تستطيع أن تجعل اللعبة
محتوية على عدّاد يقوم بحساب عدد
المحاولات ؟

إن هذا يتطلب استخدام متغير
جديد تزيد قيمته بمقدار 1 كلما
كلما أدخلنا رقماً جديداً .

وبعد إضافة مثل هذا التعديل إلى البرنامج فإن أحد الصور المقبولة له
تكون كالآتي حيث يكتب رقم المحاولة أمام العلامة ">" باستمرار :

GUESSNUMBER2

I AM THINKING OF A NUMBER BETWEEN 0
AND 100. SEE IF YOU CAN GUESS IT.

1 > 50

TOO HIGH

2 > 25

TOO LOW

3 > 35

TOO HIGH

4 > 30

TOO HIGH

5 > 28

GOT IT!

رقم المحاولة

حاول .. واستمتع بالنتائج .

من أجمل الألعاب التي يمكن أن نستمتع بها مع الكمبيوتر هي الألعاب التعليمية لأنها دائماً تأتي بالجدید ، فهي لا تفقد عنصر الإثارة أبداً . بينما مع ألعاب القذائف والحروب عادة ما تبدأ اللعبة مثيرة للحماس والرغبة في تحطيم الرقم المسجل من أحد الأصدقاء المهرة لكنك بمجرد أن تتدرب عليها وتستطيع تسجيل رقم قياسي تبدأ في البحث عن غيرها إذ تفقد الإثارة والتشويق .

والألعاب التعليمية تصل إلى ذروتها في الإبداع والإثارة عندما تجمع بين الهدف التعليمي وبين الإبداع الفني في تكوين « قصة » اللعبة وتدعيمها بالرسم الجميل والألوان والمؤثرات الصوتية .

ولا نتوقع بالطبع أن نقدم في كتاب تعليم لغة لوجو كل المهارات التي تجعلنا نصمم لعبة كومبيوترية كاملة بلغة لوجو . فنحن هنا نقدم المنطق الأساسي للعبة أما الرسم والتلوين والموسيقى والتحويلات المختلفة التي تجعل منها لعبة منافسة للألعاب الكومبيوترية الأخرى فكل هذا يحتاج منا المزيد من المهارة والتدريب على بناء الألعاب الكومبيوترية . ولعلنا في لقاء قريب إن شاء الله نلتقي حول برمجة الألعاب بلغة لوجو كما التقينا من قبل حول برمجة الألعاب بلغة بيسك .

واللعبة التي نحن بصددنا الآن عبارة عن أسئلة في الرياضة يطرحها عليك الكمبيوتر ويصحح لك الإجابات ويمنحك درجة في النهاية .

هذا هو أساس اللعبة .

أما الإضافات التي يمكن أن نضيفها إليها فلا حصر لها .

فبدلاً من الأرقام المجردة يمكن استخدام مركبات الفضاء التي تغزو الأرض من الفضاء الخارجي والتي لا يمكن إصابتها إلا بضربها بمدافع « الأجوبة الصحيحة » !

وبصفة عامة فأياً كان شكل اللعبة فهي تعتمد على منطق أساسى فى البناء .
وهذا هو الذى نقدمه لك الآن .

● تشغيل اللعبة :

عندما تكتب اسم اللعبة على الشاشة فإنها تُمسح وترى على
الشاشة السؤال الآتى :

HOW MANY PROBLEMS WOULD YOU LIKE ?

ولك أن تحدد عدد الأسئلة التى يتضمنها الامتحان فيبدأ بعدها
الكمبيوتر فى تقديم واحداً تلو الآخر .
والآتى بعد مثال للحوار الدائر على الشاشة :

MATHQUIZ

HOW MANY PROBLEMS WOULD YOU LIKE?

2

PROBLEM 1

$$35 + 41 = 76$$

CORRECT

PLEASE PRESS THE ENTER KEY

PROBLEM 2

$$78 + 43 = 111$$

SORRY, THE ANSWER IS 121

PLEASE PRESS THE ENTER KEY

YOUR SCORE IS 1

OUT OF 2 PROBLEMS

تحديد عدد الأسئلة

الحل

تقييم الكمبيوتر

الحل

.. الكمبيوتر .. الإجابة خطأ

نتيجة الامتحان

● بناء البرنامج :

بهذه الصورة السابقة للبرنامج يمكن أن نتصور أنه مبنى بالأسلوب الآتى :

- ١ - اختيار عدد المسائل المطلوب حلها .
- ٢ - اختيار عددين لتمثيل المسألة الأولى .

٣ — اختبار الإجابة وطبع نتيجة الاختبار CORRECT أو SORRY مع زيادة عدد الإجابات الصحيحة بمقدار 1 في حالة الأولى .

٤ — اختبار بلوغ عدد الأسئلة إلى الحد المطلوب ، فإذا كان كذلك يتم طبع النتيجة النهائية وينتهي البرنامج . وإلا فيتم زيادة العدد بمقدار 1 واختبار رقمين جديدين .

● البرامج :

يمكن للأصدقاء المتعجلين بلوغ النتائج كتابة مجموعة البرامج الآتية والبدء في تجربة اللعبة فوراً :

TO MATHQUIZ
 CLEARTEXT
 GETTOTAL
 MAKE "COUNT1
 MAKE "SCORE 0
 ADDQUIZ :COUNT :TOTAL :SCORE
 END

البرنامج الرئيسي

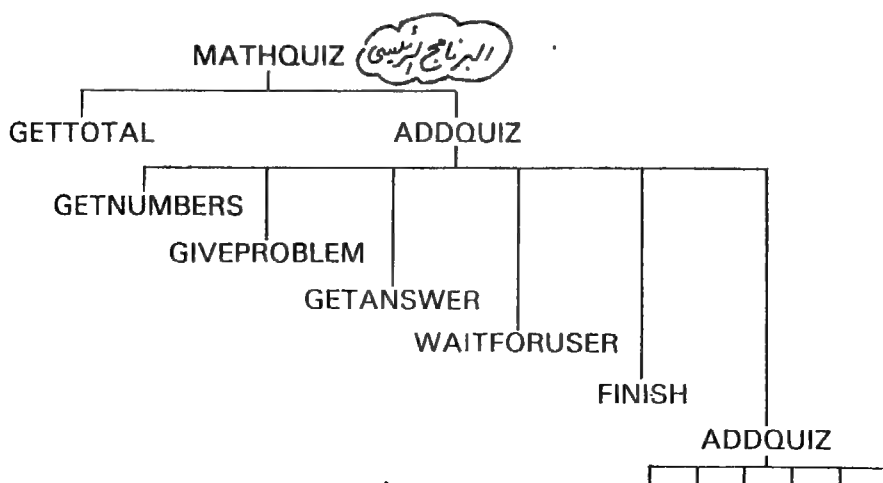
- 1 TO GETTOTAL
 PRINT [HOW MANY PROBLEMS DO YOU WANT?]
 MAKE "TOTAL READNUMBER
 END
- 2 TO ADDQUIZ :COUNT :TOTAL :SCORE
 CLEARTEXT
 GETNUMBERS
 GIVEPROBLEM
 GETANSWER
 WAITFORUSER
 IF :COUNT = :TOTAL [FINISH STOP]
 ADDQUIZ (:COUNT + 1) :TOTAL :SCORE
 END
- 3 TO GETNUMBERS
 MAKE "NUMBER1 RANDOM 100
 MAKE "NUMBER2 RANDOM 100
 MAKE "RIGHTANSWER :NUMBER1 + :NUMBER2
 END
- 4 TO GIVEPROBLEM
 PRINT SENTENCE [PROBLEM] :COUNT
 PRINT []
 TYPE (SENTENCE :NUMBER1 [+] :NUMBER2 [=])
 END
- *5 TO GETANSWER
 MAKE "RESPONSE READNUMBER
 TEST :RESPONSE = :RIGHTANSWER
 IFTRUE [PRINT [CORRECT]]
 IFTRUE [MAKE "SCORE :SCORE + 1]
 { IFFALSE [PRINT SENTENCE [SORRY, THE ANSWER IS]
 :RIGHTANSWER]
 END
- 6 TO WAITFORUSER
 PRINT [PLEASE PRESS THE ENTER KEY]
 PRINT READLIST
 END
- 7 TO FINISH
 CLEARTEXT
 PRINT SENTENCE [YOUR SCORE IS] :SCORE
 PRINT (SENTENCE [OUT OF] :TOTAL [PROBLEMS])
 END

البرامج الفرعية

● ملاحظة : البرنامج رقم 5 قد لا يعمل على بعض أجهزة الكمبيوتر ولذلك سوف نستبدله ببرنامج فرعى آخر أثناء المناقشة التالية :

● مناقشة اللعبة :

يمكن تصور اللعبة كشجرة من البرامج الفرعية تخرج جميعاً من البرنامج الرئيسى MATHQUIZ وتتسلسل كما في الشكل التالى :



* ولنبدأ بالبرنامج الرئيسى MATHQUIZ :

يبدأ البرنامج بتنظيف الشاشة ثم يستدعى البرنامج الفرعى GETTOTAL لكى يسألك عن عدد الأسئلة المطلوبة فى الامتحان .
وبإدخال عدد الأسئلة يخزنها البرنامج الفرعى GETTOTAL فى المتغير TOTAL .

ثم يقوم البرنامج الرئيسى بوضع قيمة ابتدائية (0) لعداد النتيجة SCORE ،
وقيمة ابتدائية (1) لعداد الأسئلة COUNT . وهذه المتغيرات الثلاثة يتم استدعاء البرنامج الفرعى ADDQUIZ .

```

TO MATHQUIZ
CLEARTEXT
GETTOTAL
MAKE "COUNT 1
MAKE "SCORE 0
ADDQUIZ :COUNT :TOTAL :SCORE
END
TO GETTOTAL
PRINT [HOW MANY PROBLEMS DO YOU WANT?]
MAKE "TOTAL READNUMBER
END

```

- * أما البرنامج ADDQUIZ فهو يستخدم المتغيرات الثلاثة المذكورة كأدلة له .
- وهو يبدأ بمسح الشاشة ثم يستدعي البرامج الأربعة الآتية واحداً تلو الآخر :
- ١ — GETNUMBERS لاختيار الأعداد للمسائل وتحديد الإجابة الصحيحة .
 - ٢ — GIVEPROBLEM لطبع المسألة على الشاشة .
 - ٣ — GETANSWER لاستقبال الإجابة من مستخدم البرنامج واختبارها .
 - ٤ — WAITFORUSER لإتاحة بعض الوقت للاعب قبل أن ينتقل إلى المسألة التالية .

بعد استدعاء هذه البرامج واحداً تلو الآخر فإن البرنامج ADDQUIZ يقوم باختبار قيمة المتغير COUNT لمعرفة ما إذا كان عدد الأسئلة قد بلغ غايته .

```

TO ADDQUIZ :COUNT :TOTAL :SCORE
CLEARTEXT
GETNUMBERS
GIVEPROBLEM
GETANSWER
WAITFORUSER
IF :COUNT = :TOTAL [FINISH STOP]
ADDQUIZ ( :COUNT + 1 ) :TOTAL :SCORE
END

```

فإذا كان العداد COUNT يحتوى على قيمة مماثلة لقيمة المتغير TOTAL يتفرع البرنامج إلى البرنامج الأخير FINISH أو يضاف واحد إلى العداد COUNT ويستدعى البرنامج ADDQUIZ نفسه من جديد لحل مسألة جديدة .

* وهذا هو البرنامج الفرعى GETNUMBERS :

```
TO GETNUMBERS
MAKE "NUMBER1 RANDOM 100
MAKE "NUMBER2 RANDOM 100
MAKE "RIGHTANSWER :NUMBER1 + :NUMBER2
END
```

فهو يبدأ بتخصيص عددين عشوائيين NUMBER1 , NUMBER2 ومنهما يستخرج الإجابة الصحيحة لعملية الجمع RIGHTANSWER .

* أما البرنامج GIVEPROBLEM فهو يتسلسل كالآتى :

```
TO GIVEPROBLEM
PRINT SENTENCE [PROBLEM] :COUNT
PRINT [ ]
TYPE (SENTENCE :NUMBER1 [+ ] :NUMBER2 [=])
END
```

طباعة المسألة
عن البشاشة

يطبع على الشاشة الجملة المكونة من الكلمة PROBLEM يتبعها رقم المسألة COUNT .

بلى ذلك طباعة سطر خال بالأمر [] PRINT .

ثم يقوم البرنامج بطباعة المسألة نفسها وهى عبارة عن مجموع العددين :
NUMBER2 ، NUMBER1 .

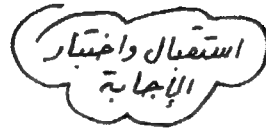
* وهذا هو البرنامج الفرعى GETANSWER الذى يستقبل إجاباتك :

* TO GETANSWER

```

1 MAKE "RESPONSE READNUMBER
2 TEST :RESPONSE = :RIGHTANSWER
3 IFTRUE [PRINT [CORRECT]]
4 IFTRUE [MAKE "SCORE :SCORE + 1]
5 | IFFALSE [PRINT SENTENCE [SORRY, THE ANSWER IS]
  | :RIGHTANSWER]
END

```



يستقبل هذا البرنامج إجابتك مستخدماً البرنامج الفرعي الخاص READNUMBER الذى يجب تحميله من القرص المغنطيسى (LWIL) أو كتابته بالطريقة التى قدمناها فى اللعبة السابقة ضمن البرامج الفرعية . وبعد استقبال الإجابة يتم تخصيصها للمتغير . RESPONSE كما فى السطر 1 .

ويقوم البرنامج باختبار وصحة الإجابة باستخدام الأمر الجديد TEST الذى يستخدم جنباً إلى جنب مع كل من الأمرين IFTRUE ، IFFALSE لتكوين عملية شرطية متكاملة .

ويستخدم الأمر TEST متبوعاً بالشرط المطلوب اختبار صحته وهو هنا شرط تساوى الإجابة المدخلة وهى RESPONSE بالإجابة الصحيحة وهى RIGHTANSWER (كما فى السطر 2) .

ثم يأتى السطر الثالث ليقول « إذا كان الاختبار صحيحاً افعل كذا وكذا .. »

IFTRUE

ونرى فى السطر الثالث الأمر IFTRUE يتبعه أمر طباعة الكلمة . CORRECT

ثم نرى فى السطر الرابع نفس الأمر يتبعه الأمر MAKE الذى يستخدم هنا لزيادة رصيد النتيجة بمقدار واحد نظراً لصحة الإجابة .

أى أن الأمر (أو الأوامر) IFTRUE يعقبا كل ما تريد تنفيذه عندما تكون النتيجة صحيحة (TRUE) .

والأمر IFFALSE فى السطر الخامس يعقبه ما نريد تنفيذه عندما لا يتحقق الشرط أى عندما تكون نتيجة المقارنة غير صحيحة (FALSE) . ونرى فى السطر الخامس أنه يتم طباعة الرسالة "SORRY...." يعقبا طباعة الإجابة الصحيحة للسؤال .

والأمر TEST قد لا يتوفر ببعض أجهزة الكمبيوتر لذلك يمكننا بناء نفس منطق هذا البرنامج باستخدام الأمر IF الذى نعرفه فيصبح البرنامج كالآتى :

```
TO GETANSWER
MAKE "RESPONSE READNUMBER
IF "RESPONSE = :RIGHTANSWER IPR
"CORRECT MAKE "SCORE :SCORE + 11
[PR SE [SORRY, THE ANSWER IS] :
RIGHTANSWER]
END
```

صورة أخرى للبرنامج (GETANSWER)

* بعد ذلك نأتى إلى البرنامج WAITFORUSER بمعنى « انتظر المستخدم » وكل ما يفعله هذا البرنامج هو طباعة الرسالة « اضغط على الزر ENTER » .

وبعد هذه الرسالة يقوم الأمر READLIST بتعليق البرنامج حتى يقرر المستخدم أن ينتقل للسؤال التالى وبالضغط على الزر ENTER فإن الأمر READLIST يستقبل القائمة الخالية [] ويطبعها ومع ذلك يمكنك كتابة أى شىء قبل الضغط على الزر ENTER .

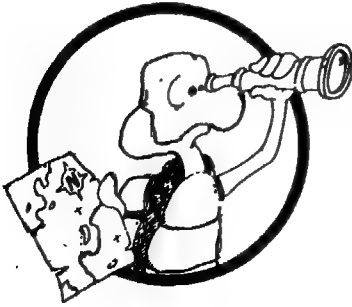
```
TO WAITFORUSER
PRINT [PLEASE PRESS THE ENTER KEY]
PRINT READLIST
END
```

البرنامج :
انتظر المستخدم

● برنامج النهاية FINISH :

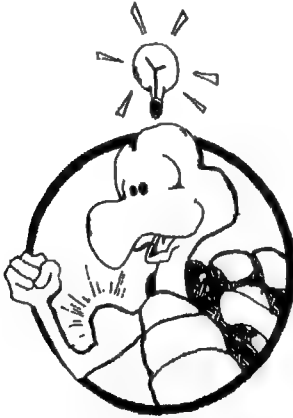
يبدأ البرنامج بمسح شاشة الكتابة . ثم يطبع النتيجة النهائية SCORE متضمنة عدد الأجوبة الصحيحة وعدد الأسئلة الكلي .

تمرين



- يمكنك تعديل البرنامج السابق لكي يقرر اللاعب حدود الأرقام التي سوف يستخدمها في الامتحان فالبرنامج يمكن أن يوجه للأطفال الصغار وفي هذه الحالة يمكن أن يحتوى على جمع الأرقام البسيطة التي لا تتعدى العشرة . كما يمكن أن يحتوى البرنامج على عمليات حسابية كبيرة .
- لعله من المناسب أن يحتوى البرنامج على عدّة مستويات من الصعوبة وكلما نجح اللاعب في أحد المستويات انتقل إلى المستوى الأعلى ..
- هل تستطيع أن تطبع رسائل مختلفة عقب كل إجابة سواء في حالة الإجابة الصحيحة أو الخاطئة ؟
- اكتب التعديل اللازم لتحويل اللعبة إلى امتحان في الطرح ثم آخر في القسمة أو الضرب .
- هل يمكنك تعميم البرنامج ليشمل العمليات الحسابية جميعاً ؟
- أحد طرق التنفيذ هي أن يختار اللاعب نوع العملية الحسابية (جمع أو طرح أو ضرب أو قسمة) .

فكرة



يمكن تطوير اللعبة بطريقة مثيرة
للخيال إذا استخدمت السلاحف في
تقديم المسائل وطباعة النتائج .. إنه
مشروع يحتاج إلى التعاون بين
مجموعة من الأصدقاء من عشاق لغة
لوجو ..

ASCII

● الكود آسكى لكل زر من الأزرار

كما نعلم أن كل زر تضغط عليه من لوحة الأزرار ينقل إلى الكمبيوتر رقماً
كودياً معيناً يمثل اللبنة المكتوبة على الزر .

كما أن بعض أزرار الكمبيوتر لا تكتب شيئاً مثل زر المسافة الخالية وأزرار
التحكم الخاصة بإيقاف البرنامج أو الخروج من المحرر .. إلخ .

وجميع هذه الوظائف واللبنتات ممثلة بأرقام كودية في جدول قياسى عالمى
يسمى جدول الكود آسكى (ASCII code) وهو موضح بالملحق رقم (٤) .

وهذا الجدول يسمح بتمثيل أكواد مختلفة حتى الرقم 255 ومع ذلك فعدد
الأكواد به أقل من ذلك . وهذا يعطى شركات الكمبيوتر الفرصة لابتكار
أشكال مختلفة للرسم أو حروف جديدة أو وظائف جديدة للأزرار وتكويدها
بداخل الجدول . وبذلك فإن كل كومبيوتر له جدولته الخاص للكود الذى
يتضمن الشفرة العالمية علاوة على الشفرات الخاصة له .

ولمعرفة كود أى لبنة من اللبئات المكتوبة على الأزرار نستخدم أمر العملية
آسكى كالآتى :

PRINT ASCII "A ← كود الحرف A

65 ← الجواب

PRINT ASCII "B

66 ← كود الحرف B

● الموسيقى بلغة لوجو **SOUND**

تتوفر المؤثرات الصوتية بلغة لوجو بالأجهزة المختلفة بطريقة مشابهة
للمؤثرات الصوتية فى لغة بيسك . ومن خصائص الموسيقى بصفة عامة أنها
تعتمد كثيراً على طراز الكمبيوتر ، لذلك يوصى بالرجوع إلى كتاب
الاستخدام للغة للكمبيوتر المعين .

وبصفة عامة فإن أمر الموسيقى عادة يأخذ مُدخلين هما الفترة الزمنية لعزف
النغمة وتردد النغمة .

والآتى بعد مثال لأمر الموسيقى بالكمبيوتر المنزلى سنكلىر لتوضيح معنى
المُدخلات المستخدمة معه :

SOUND [x y]

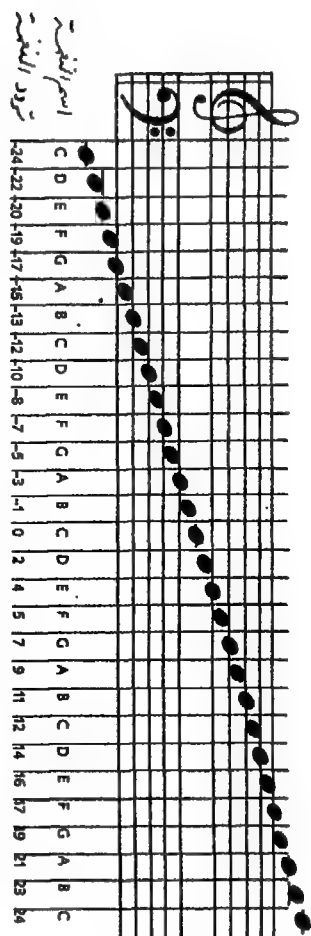
حيث : x

هو زمن النغمة مقاساً بالثوانى ويتراوح بين 10,0

y :

هو النغمة (تردد النغمة) وهى رقم يتراوح بين 60 - ، 69 .

والشكل الآتى يوضح السلم الموسيقى لجهاز سنكلىر والقيمة العددية لكل
نغمة على مدى ٤ أوكتاف علماً بأن القيمة العددية لنغمة دو الوسطى هى 0 ،



دوال وسطى

والبرناج: التالى يحول أزرار الكومبيوتر سنكلير إلى أزرار بيانو :

TO SING

SOUND SE 0.5 (ASCII RC) - 56

SING

END

ملاحق الكتاب

الملحق رقم ١ :

الروتينات الفرعية لرسم دائرة ذات نصف قطر معين R (للكومبيوتر IBM)

نناقش هنا كيفية رسم دائرة ذات نصف قطر معين وهو نفس المبدأ الذي تقوم عليه البرامج الفرعية الجاهزة LCIRCLE, RCIRCLE وأيضاً البرامج الفرعية للأقواس LARC, RARC (للكومبيوتر IBM) وفيما يلي نص هذه البرامج الموجودة على القرص "LWIL" الذي أشرنا إليه من قبل :

```
TO RCIRCLE :R  
REPEAT 36 [RCP :R]  
END
```

دائرة يمينى

```
TO RARC :R  
REPEAT 9 [RCP :R]  
END
```

قوس / يمين

```
TO RCP :R  
RIGHT 5  
FORWARD :R * ( 3.14159 ) / 18  
RIGHT 5  
END
```

```
TO LCIRCLE :R  
REPEAT 36 [LCP :R]  
END
```

دائرة يسرى

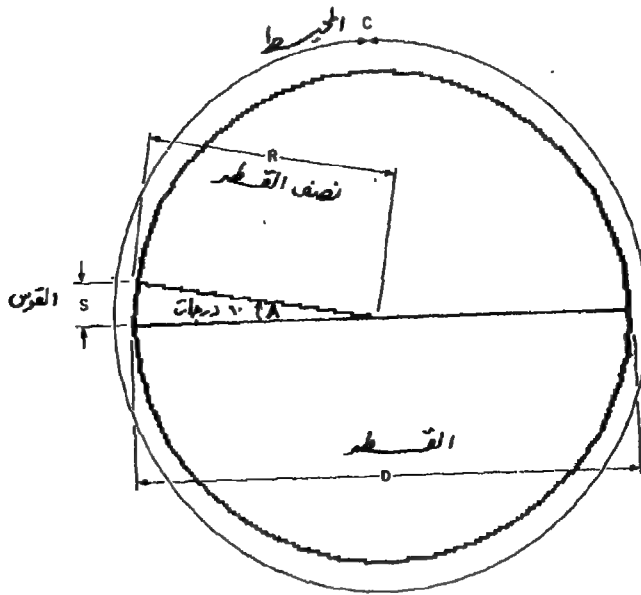
```
TO LARC :R  
REPEAT 9 [LCP :R]  
END
```

قوس / يسرى

```
TO LCP :R  
LEFT 5  
FORWARD :R * ( 3.14159 ) 18  
LEFT 5  
END
```

ولنبداً هنا بمناقشة الروتين R: RCIRCLE لرسم دائرة نصف قطرها R تقع في الجانب الأيمن من الشاشة . وكما نرى أنه برنامج يعتمد على برنامج فرعى آخر هو R: RCP الذى يستقبل نصف القطر R ويقوم برسم الدائرة باستخدام الأمرين FORWARD , RIGHT .

والدائرة التى نقوم برسمها نقسمها إلى ٣٦ جزءاً أى يتم تقسيم الزاوية ٣٦٠ درجة كلها إلى ٣٦ زاوية صغيرة مقدار كل منها ١٠ درجات . وبذلك فإن طول القوس المواجه لكل ١٠ درجات عبارة عن جزء من ٣٦ جزءاً من محيط الدائرة .



هذا القوس الصغير سوف نعتبره طول الخطوة التى تتحركها السلحفاة أثناء رسمها للدائرة أى أنها سوف تتحرك ٣٦ خطوة وتدور ١٠ درجات فى كل مرة وبذلك تقطع الرحلة الكلية الواقعة فى ٣٦٠ درجة . فما طول القوس S (انظر الرسم) ؟

$$C = 2 * 3.14159 * R$$

إن محيط الدائرة هو

$$S = C/36 = 2 * 3.14159 * R/36$$

والقوس هو

وهذا القوس هو الذى نستخدمه مع الأمر FD باعتباره خطأ مستقيماً (تقريباً) .

$$FD :R*(3.14159)/18$$

* ملاحظة ١ :

هذه هي القيم المستخدمة في البرنامج RCIRCLE للكمبيوتر IBM ولك الحرية أن تقسم الدائرة إلى ما تشاء من الأجزاء بدلاً من ٣٦ جزءاً .

ويمكن اعتبار الصيغة العامة كالآتي :

$$S = C/N$$

$$S = 2 * 3.14159 * R/N$$

أى :

حيث N عدد الأجزاء .

* ملاحظة ٢ :

يمكن أيضاً استنتاج طول القوس باستخدام الدوال المثلثية :

$$S = R * \sin A$$

حيث A زاوية الدوران .

حتى الآن يمكننا كتابة أمر الدائرة كالآتي :

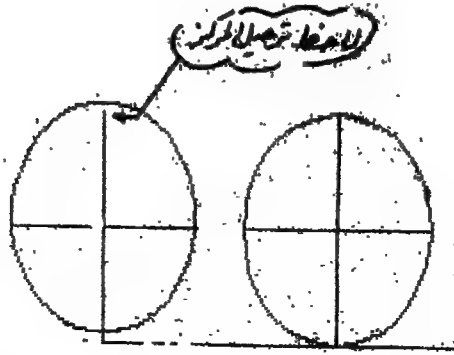
REPEAT 36 [FD :S RT 10] ١

ولكن البرنامج RCP يقسم الزاوية إلى نصفين كالآتي :

REPEAT 36 [RT 5 FD :S RT 5] ٢

● وهذه فصة أخرى تتعلق بالدقة :

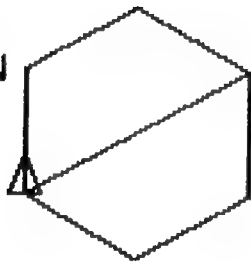
فالأمر رقم ١ لا يجعل مركز الدائرة في المنتصف تماماً بل يجعله أسفل موضعه الأصلي قليلاً ورغم أن هذا لن يكون ملحوظاً مع الدوائر الصغيرة لكنه يظهر كلما زاد حجم الدائرة والآتي بعد شكل يوضح دائرتين إحداهما مرسومة بالأمر ١ والأخرى بالأمر رقم ٢ وتم توصيل خطوط مستقيمة بينهما لتوضيح العيوب المحتملة في الرسم .



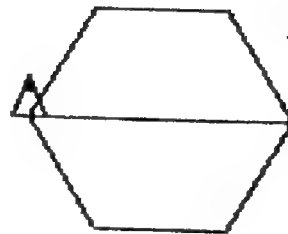
ويظهر هذا العيب مجسماً عندما نرسم مضلعاً ذي عدد قليل من الأضلاع (باعتبار أن الدائرة مضلع ذو أضلاع كثيرة) وجرب المثالين الآتيين :

REPEAT 6 [FORWARD 50 RIGHT 60]

REPEAT 6 [RIGHT 30 FORWARD 50 RIGHT 30]



(P)



(B)

فبالمثال الأول الذى يرسم الشكل (أ) نلاحظ أن نقطة البداية تقع أسفل مركز الشكل .

أما فى المثال الثانى الذى يعطى الشكل (ب) فتقع نقطة البداية فى المنتصف تماماً .

لذلك تعتبر هذه الملاحظة ملاحظة عامة لكل الأشكال المضلعة .

أما الأقواس RARC ، LARC فهى تعتمد على نفس المبدأ ولكن عدد مرات التكرار تختلف فهى ٩ مرات بدلاً من ٣٦ مرة أى ربع عدد المرات .

الملحق رقم ٢ :

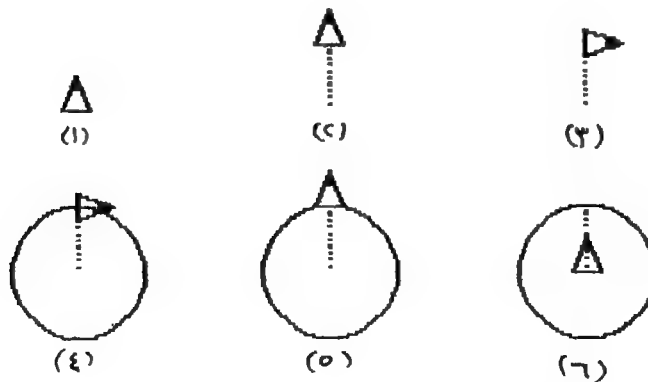
رسم دائرة فى منتصف الشاشة

الروتين الفرعى CCIRCLE

(للكومبيوتر IBM)

أما البرنامج الفرعى لرسم الدائرة فى منتصف الشاشة فهو يتضمن تحريك نقطة البداية إلى أعلى بمقدار نصف القطر R ثم إدارة السلحفاة يميناَ بحيث ترسم الدائرة فى الجزء السفلى فتظهر فى منتصف الشاشة وهذا هو الجزء الرئيسى فى البرنامج ويليه خطوات تنفيذ البرنامج :

HIDETURTLE
 PENUP FORWARD :R ← (١) (٢)
 RIGHT 90 ← (٣)
 PENDOWN REPEAT 36 [RCP :R] ← (٤)
 LEFT 90 ← (٥)
 PENUP BACK :R ← (٦)



الملحق رقم ٣

البرنامج الفرعي لقراءة الأعداد EADNUMBER

(أحد البرامج الفرعية المساعدة على القرص (LWIL procedures) للكمبيوتر
 (IBM

```
TO READNUMBER
MAKE "NUM1 READLIST
TEST :NUM1 = [ ]
IFTRUE [PRINT [PLEASE TYPE A NUMBER] OUTPUT READNUMBER]
TEST NOT NUMBERP FIRST :NUM1
IFTRUE [PRINT [PLEASE TYPE A NUMBER] OUTPUT READNUMBER]
IFFALSE [OUTPUT FIRST :NUM1]
END
```

● ملاحظة :

يحتوى هذا البرنامج على العملية الشرطية TEST التى ربما لا تتوفر مع بعض الأجهزة وفى هذه الحالة يمكن إنشاء نفس البرنامج الفرعى بطريقة أخرى باستخدام العملية الشرطية IF وحفظه على القرص المغنطيسى بنفس الاسم READNUMBER وقد قدمنا مثل هذا البرنامج فى الباب التاسع .

● فكرة البرنامج :

يستخدم هذا البرنامج لاستقبال الأعداد من لوحة الأزرار بطريقة مماثلة للأمر READLIST الذى يستخدم لاستقبال القوائم .

وللتوضيح جرب المثال الآتى :

```
MAKE "NUMBER READLIST
55
PRINT :NUMBER
55
```

فالعدد 55 تم استقباله من لوحة الأزرار بالأمر READLIST وتمت طباعته أيضاً . وحتى الآن يبدو كل شئ على ما يرام .

ولكن حقيقة الأمر أن هذا العدد ليس إلا قائمة بمعنى أنه لا يجوز إدخاله فى أية عمليات حسابية .

وللتأكد من ذلك نجرب المثال الآتى :

```
PRINT (:NUMBER = 55)
FALSE
```

إذن فالعدد 55 ليس رقماً ..

أيضاً عند محاولة جمعه على رقم ما فإن الكمبيوتر يعطى رسالة خطأ ولو أنك كتبت الأمر الآتى :

PRINT :NUMBER + 5

فإنك تستقبل الرسالة التالية من الكمبيوتر :

“+ does'nt like [55] as input”

ولكن يمكن الحصول على العدد 55 من هذه القائمة بطباعة أول كلمة فيها بالأمر الآتي :

PRINT (FIRST :NUMBER)

55

وللتأكد من أن العدد 55 الناتج بهذه الطريقة هو عدد فعلاً أدخل الأمر الآتي لتحصل على النتيجة TRUE :

PRINT (FIRST :NUMBER) = 55

TRUE

وهدف البرنامج REMDNUMBER هو استقبال العدد وإرساله بالأمر OUTPUT إلى برنامج آخر .

لذلك يمكن كتابة البرنامج ببساطة كالآتي :

**TO READNUMBER
OUTPUT FIRST READLIST
END**

هذا هو صلب البرنامج . ويمكن استخدامه بأمر مباشر أو من خلال برنامج . بمعنى أنه طالما يوجد هذا البرنامج معك في حيز العمل يمكنك إعطاء أمر مثل :

PRINT READNUMBER + 5

عندئذ سوف ينتظر منك البرنامج أن تدخل رقماً ما ثم يجمع عليه الرقم 5 ويطبعه على الشاشة .

أما بقية البرنامج READNUMBER الذى قدمناه بداية فيشمل بعض الشروط التى من شأنها تجنب رسالات الخطأ الناجمة عن سوء الاستخدام وهى الضغط على الزر ENTER بدلاً من إدخال رقم أو كتابة حروف أبجدية .

شفرة الكود آسكى القياسية

Character	Code	Character	Code
A	65	0	48
B	66	1	49
C	67	2	50
D	68	3	51
E	69	4	52
F	70	5	53
G	71	6	54
H	72	7	55
I	73	8	56
J	74	9	57
K	75	+	43
L	76	-	45
M	77	/	47
N	78	*	42
O	79	↑	94
P	80	(40
Q	81)	41
R	82	<	60
S	83	>	62
T	84	=	61
U	85	?	63
V	86	\$	36
W	87	"	34
X	88	,	44
Y	89	.	46
Z	90	,	59
		Carriage Return	13
		Line Feed	10
		Space	32

الملحق رقم ٥ :

إجابات مختارة



إجابات الباب الرابع

(١)

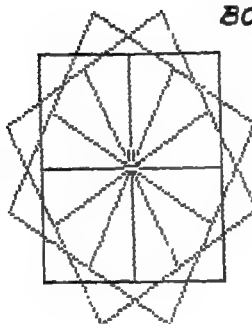
```
TO BOX4
REPEAT 4 [REPEAT 4 [fd 40 lt 90]
RT 90]
END
```

```
TO BOX3X4
REPEAT 3 [box4 rt 30]
END
```

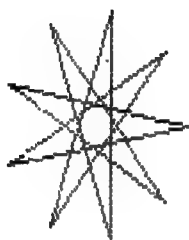
تنفيذ البرنامج BOX4



تنفيذ البرنامج
BOX3X4

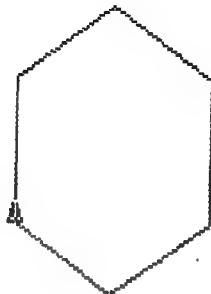


(r)



```
TO STR9
REPEAT 9 LFD 80 RT ( 360 / 9 ) *
41
END
```

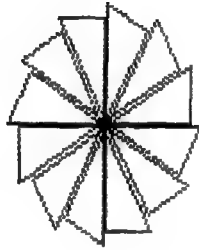
(r)



```
TO POLY6
REPEAT 6 LFD 50 RT ( 360 / 6 ) *
41
END
```

YXZ

(٤)



```
TO SILLY1
HT REPEAT 12 [SILLY]
END
```

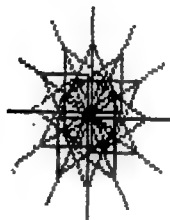
البرنامج الرئيسي

```
TO SILLY
FD 50 LT 90
FD 20 LT 120 FD 30
END
```

البرنامج الفرعي

كما يمكن بالإضافة التالية أن نرسم الشكل التالي :

```
?
TO SILLY2
HT
REPEAT 12 [SILLY RT 60 BK 50]
END
```



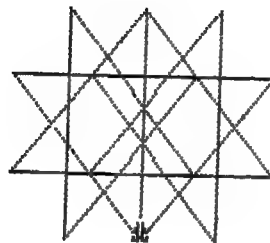
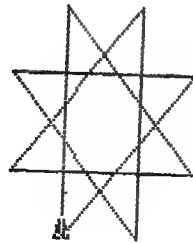
النجمة على اليسار

(٥)

```
TO STR8L
REPEAT 8 [FD 80 LT (360/8)*3]
END
```

النجمة على اليمين

```
TO STR8R
REPEAT 8 [FD 80 RT (360/8)*3]
END
```



النجمة وصورتها

إجابات الباب السادس

(١)

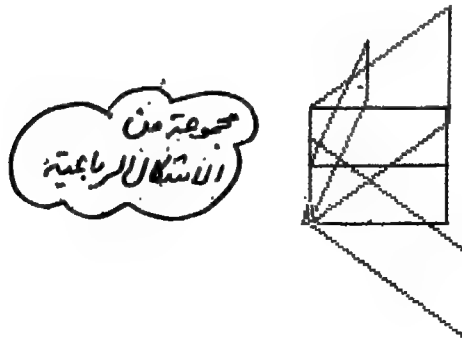


```
?
?
TO STRS
CS HT
REPEAT 5 [STARS 5 + RANDOM 50]
END
```

```
?
TO STARS :SIZE
REPEAT 5 [FD :SIZE RT 144]
PU
LT 60 FD RANDOM 60
PD
END
```

(٢)

?
TO RECTG :L :W :ANGLE
REPEAT 2 [FD :W RT :ANGLE FD :L
RT 180 - :ANGLE]
END



(٣)

?
TO VAR52
MAKE "AGE 8
MAKE "NAME [SALLY OSSAMA]
MAKE "TEL "65372
MAKE "TELEPHONE :TEL
PR [:SE [NAME:] :NAME [AGE:] :AG
E :TEL:] :TEL)
END

NAME SALLY OSSAMA AGE: 8 TEL: 6
5372

إجابات الباب الثامن



بإضافة هذا التعديل إلى البرنامج الفرعي HAT يمكن أن تحصل على الوضع المعدل للقبعة .

```
?
TO HAT
PU FD 35
PD DRAUHAT
PU BK 40 PD
END
```

النتيجة

الملحق [٦] :

ملخص بأوامر لوجو ألتى وردت بالكتاب حسب ترتيب ورودها

الاختصار	الأمر
PR	PRINT
CS	CLEARSCREEN
FD	FORWARD
BK	BACK
RT	RIGHT
LT	LEFT
TS	TEXTSCREEN
PU	PENUP
PD	PENDOWN
HT	HIDETURTLE
ST	SHOWTURTLE
	SETPAL*
	SETPC
	SETBG
	REPEAT
	FENCE
	WRAP
	WINDOW
	CLEAN
	HOME
	STARTROBOT
	STOPROBOT
	TO
	END
ED	EDIT
ER	ERASE
	ERALL
	PO

* الأوامر المتبوعة بالعلامة (*) خاصة بالكمبيوتر IBM أو ما يوافقه .

الاختصار	الأمر
	POALL POTS SAVE LOAD DIR* CATALOG** ERASEFILE* SAVEPIC* LOADPIC* DRIBBLE* NODRIBBLE* PRINTON** RINTOFF** COPYSscreen** OUTDEV 1*** OUTDEV 0*** RCIRCLE* LCIRCLE* RARC* LARC* TRUE FALSE SUM DIV PRODUCT EQUALP ROUND INT RANDOM

** أوامر خاصة بكمبيوتر سنكلير

*** أوامر خاصة بكمبيوتر أبل

الاختصار	الأمر
SIN	SINE
COS	COSINE
TAN	TANGENT
	ARCSIN
	ARCCOS
	ARCTAN
COT	COTANGENT
	ARCCOT
	REMAINDER
	SQRT
SE	SENTENCE
	WORD
	FIRST
BF	BUTFIRST
	LAST
BL	BUTLAST
RL	READLIST
	MAKE
SCR	SCRUNCH
SETSCR	SETSCRUNCH
	IF
	STOP
	HEADING
RC	READCHAR
	OR
	AND
OP	OUTPUT
	NUMBERP
	NOT

الاختصار	الأمر
CT	TYPE CLEARTEXT READNUMBER* TEST IFTRUE IFFALSE ASCII SOUND CCIRCLE*

صدر للمؤلف في مجال الكمبيوتر

من مكتبة ابن سينا

١ — تحدث مع الكمبيوتر بلغة كويول

... المستوى الأول

٢ — كل شيء عن الكمبيوتر (وكتابة البرامج بلغة بيسك)

... مبسط للنشء ولأولياء الأمور

٣ — تحدث إلى الكمبيوتر بلغة بيسك

... حتى المستوى المتقدم من لغة بيسك

يضم اللغة القياسية قديمها وحديثها وأيضاً

أشهر طرازات لغة بيسك .

٤ — كيف يفكر الكمبيوتر

... خرائط التسلسل المنطقي للبرامج والنظم

الآلية وتحويل النظم اليدوية إلى آلية .

٥ — برمجة الألعاب الكمبيوترية

... طرق برمجة القذائف والتصادم

والمؤثرات الصوتية مشروحة بلغة الطرازات

الشهيرة للكمبيوتر المنزلي في مصر والعالم

العربي علاوة على لغة بيسك القياسية

(ميكروسوفت) .

٦ — مدخلك إلى عالم الكمبيوتر .. المعالجة الإلكترونية للبيانات EDP

٧ — تعلم لغة الكمبيوتر سى من خلال لغة بيسك

مدخل مناسب للهواة والمحترفين

لإجادة لغة سى

٨ - الرسم بالكمبيوتر

... يتناول كل ما يخص استخدام
الكمبيوتر في الرسم .. يشرح عبارات
بيسك القياسية للرسم الدقيق علاوة على
أهم اللهجات المنتشرة للأجهزة الكمبيوتر
المتزلى .

٩ - سلسلة قصص الخيال العلمي

- ١ - إعدام إنسان آلي
 - ٢ - الدخول في الثقب الأسود
 - ٣ - المعلوم والمجهول
-

١٠ - تحدث إلى الكمبيوتر بلغة فورتران ٧٧

... مرجعك العربى فى لغة فورتران يبدأ من
البدايات الأولى للغة ويصل حتى مستويات
متقدمة فى إنشاء البرامج . يضم الكتاب كل
عبارات اللغة قديمها وحديثها مع تطبيقات
على مختلف أجهزة الكمبيوتر .

١١ - برامج وألعاب كومبيوترية مشروحة (بلغة بيسك)

... برامج تعليمية فوازير ألعاب حروب
وقذائف ومغامرات .. مقدمة بلغة بيسك
على أشهر طرازات الكمبيوتر المتزلى :
تكساس ، كومودور ، أتارى ، BBC ،
إلكترون ، سنكلير ..

فهرس الكتاب

الصفحة

الموضوع

كلمة المؤلف ٥

■ الباب الأول : لنكتشف معاً عالم لوجو :

لنتعرف بعائلة لوجو ٩

● إعداد الكومبيوتر للعمل ١٣

● لنكتب أمراً بلغة لوجو PRINT ١٤

● لا تخش أن تتلف الكومبيوتر ١٨

● هيا نلتقى بالسلحفاه CLEARSCREEN, FORWARD, BACK

RIGHT, LEFT

■ الباب الثاني : التحكم في السلحفاه :

● عالم السلحفاه TEXTSCREEN PENUP, PENDOWN ٢٧

● المزيد من الأوامر للسلحفاه ٢٩

● السلحفاه تحزم الشاشة ٣١

● لنرسم مربعاً بالسلحفاه ٣٣

● إخفاء السلحفاه وإظهارها HIDETURTLE, SHOWTURTLE ٣٥

● الرسم بالألوان SETPAL, SETPC ٣٧

● تلوين الخلفية SETBG ٣٨

● الألوان وأجهزة الكومبيوتر المختلفة ٣٩

● فلنتقدم خطوة أخرى إلى الأمام ٤٠

● ارسم مربعاً بأمر واحد فقط REPEAT ٤١

● بناء سور حول السلحفاه FENCE, WRAP ٤٢

● شاهد السلحفاه من النافذة WINDOW ٤٣

● تنظيف الشاشة بالأمر CLEAN ٤٤

● إعادة السلحفاه إلى بيتها HOME ٤٤

● التحكم في الروبوت STARTROBOT, STOPROBOT ٤٦

■ الباب الثالث : تعليم الكمبيوتر أوامر جديدة :

- ٤٩ ● عَلم الكمبيوتر أمراً جديداً TO, END
- ٥٢ ● إصلاح برنامج لوجو EDIT
- ٥٤ ● كتابة البرنامج باستخدام المحرر ERASE, ERALL
- ٥٦ ● ماذا يحدث بداخل الكمبيوتر ؟
- ٦٠ ● اكتب لغة الكمبيوتر بنفسك
- ٦٢ ● عرض البرنامج على الشاشة PO, POALL, POTS
- إمكانيات أخرى للمحرر ED, ED [...]
- ٦٣ ● حفظ البرامج في الذاكرة الخارجية واستدعاؤها
- SAVE, LOAD, DIR, ERASEFILE, SAVEPIC, LOADPIC
- تشغيل جهاز الطباعة SAVE LPT1, DRIBBLE, NODRIBBLE
- ٦٤ ● PRINTON, PRINTOFF, OUTDEV

■ الباب الرابع : تطبيقات مختلفة للرسم بالسلحفاة :

- ٦٩ ● استخدام البرامج الفرعية
- ٧٤ ● رسم المثلثات
- ٧٥ ● رسم النجوم
- ٧٨ ● هل هناك بديل للتجربة والخطأ ؟
- ٧٩ ● المرأة
- ٨٠ ● فكرة لهواة البحث والتنقيب .. فقط !
- ٨٤ ● الدوائر .. من المبادئ الأولية
- ٩٠ ● ارسم ما تشاء من الأشكال باستخدام الدوائر
- ٩٢ ● المنحنيات
- ٩٥ ● ارسم ثعباناً بالمنحنيات
- مشروعات للرسم
- ٩٩ ١ — حدائق الزهور
- ١٠٤ ٢ — الملاح والوجوه

- ٣ — حيوانات لوجو ١٠٧
- ٤ — ووسائل المواصلات أيضاً ١٠٨
- ٥ — مشروع لعبة كومبيوترية ١١٠

■ الباب الخامس : عالم الأرقام والحروف :

- البيانات في لغة لوجو ١١٥
- التعامل مع الأعداد في لغة لوجو :
- العمليات الحسابية ١١٦
- المقارنات TRUE, FALSE ١١٧
- هناك دائماً طريقة أخرى PRODUCT, SUM, DIV, ١١٩
- EQUALP ١١٩
- التقريب ROUND ١٢٠
- الحذف INT ١٢١
- الأعداد العشوائية RANDOM ١٢٣
- عالم الأعداد في لغة لوجو ١٢٤
- النسب المثلثية SIN, COS, TAN ١٢٤
- النسب المثلثية العكسية ARCSIN, ARCCOS, ١٢٤
- ARCTAN ١٢٤
- مقلوبات النسب المثلثية COT, ARCCOT ١٢٥
- باقي القسمة REMAINDER ١٢٥
- إيجاد الجذر التربيعي SQRT ١٢٥
- مثال عام : الرسم في إحداثيات عشوائية ١٢٥
- التعامل مع الكلمات والقوائم في لغة لوجو :
- ماهي كلمات لوجو logo words ١٢٦
- تحديد نهاية الكلمة ١٢٨

- القوائم في لغة لوجو logo lists ١٢٩
- توصيل كلمتين معاً WORD ١٣٠
- توصيل أكثر من كلمتين "... (WORD) ١٣٠
- توصيل الكلمات والقوائم في جملة SENTENCE ١٣٠
- توصيل العديد من القوائم والكلمات.. (SENTENCE) .. ١٣٠
- طباعة جزء من كلمة أو قائمة FIRST, BUTFIRST, ١٣٢
- LAST, BUTLAST ١٣٢
- الكمبيوتر يسأل وأنت تجيب READLIST ١٣٥
- ماذا يحدث بداخل الكمبيوتر ؟ ١٣٦
- الكمبيوتر يناقش (مثال) ١٤٠
- الكمبيوتر موافق دائماً (مثال) ١٤٠

■ الباب السادس : استخدام المتغيرات :

- ما معنى المتغير MAKE ١٤٥
- اختر الاسم المناسب للمتغير ١٤٥
- الكلمات والقوائم في متغيرات أيضاً ١٤٦
- تخصيص متغير لمتغير آخر ١٤٨
- طباعة العناوين ١٤٩
- الدوال functions ١٥٠
- ارسم مثلثاً متغير الضلع ١٥٢
- نداء الدالة لنفسها recursion ١٥٣
- مازلنا نلعب بالمثلثات المتغيرة ١٥٤
- التكرار مع البرامج أيضاً ١٥٨
- ارسم نجمة خماسية بأحجام مختلفة ١٥٩
- ماذا يحدث بداخل الكمبيوتر ؟ ١٦٠

- الفرق بين المتغير العام ومتغير الدالة ١٦٤
- الدالة بأكثر من دليل ١٦٦
- تطبيقات
- ١ — أشكال زخرفية بالدوائر ١٦٩
- ٢ — رسم دائرة ذات نصف قطر معين ١٧١
- ٣ — رسم دائرة في منتصف الشاشة .. ١٧٤
- ٤ — رسم مجموعة دوائر ذات مركز واحد ١٧٧
- ٥ — الأشكال الحلزونية ١٧٨
- ضبط الدوائر على الشاشة SCRUNCH, SETSCRUNCH
- ١٨٠

■ الباب السابع : مشروعات للرسم :

- لرسم شخصاً ١٨٥
- قافلة من السيارات ٢٠٠
- منطقة سكنية ٢٠٥

■ الباب الثامن : التكرار واتخاذ القرار :

- الحلقات التكرارية ٢١١
- كيف نوقف الحلقة التكرارية IF, STOP ٢١٢
- التحكم في وضع السلحفاة HEADING ٢١٥
- ماذا يحدث بداخل الكمبيوتر ؟ ٢١٨

■ الباب التاسع : بناء البرامج والألعاب الكمبيوترية :

- تغيير وظائف الأزرار READCHAR ٢٢٩
- الكلمات والقوائم كمتغيرات
- ماذا يحدث بداخل الكمبيوتر ؟ ٢٣١

- ٢٣٤ أذ ووصل القوائم والكلمات
- يحدث بداخل الكمبيوتر ؟
- ٢٤٠ ● استخدام القوائم في ألعاب الكلمات
- ٢٤٣ ● العمليات المنطقية OR, AND
- ٢٤٧ ● تمرير البيانات من برنامج إلى آخر OUTPUT
- ٢٤٨ ● اختبار الأعداد NUMBERP
- ٢٤٩ ● النفي المنطقي NOT
- ● لعبة الرقم السري TYPE, CLEARTYPE, السري
- ٢٥٠ READNUMBER
- ● لعبة- تعليمية : امتحان في الرياضة TEST, IFTRUE,
- ٢٥٩ IFFALSE
- ٢٦٩ ● الكود آسكي لكل زر من الأزرار ASCII
- ٢٧٠ ● الموسيقى بلغة لوجو SOUND

■ ■ ملاحق الكتاب :

● الملحق ١

الروتينات الفرعية لرسم دائرة ذات نصف قطر معين للكمبيوتر

LCIRCLE, RCIRCLE (IBM)

٢٧٢

LARC, RARC

● الملحق رقم ٢

٢٧٦ رسم دائرة في منتصف الشاشة (للكمبيوتر IBM) CCIRCLE

● الملحق رقم ٣

٢٧٧ البرنامج الفرعي لقراءة الأعداد (للكمبيوتر IBM) READNUMBER

• الملحق رقم ٤

٢٨٠ شفرة الكود آسكى القياسية .

• الملحق رقم ٥

٢٨١ إجابات مختارة للتمارين .

• الملحق رقم ٦

٢٩٠ ملخص بأوامر لوجو التى وردت بهذا الكتاب .

★★★

رقم الإيداع بدار الكتب ١٩٨٧/٧٩٣٧

الترقيم التولي ٠ - ٢٠ - ١٣٤١ - ٩٧٧

دار النضر للطباعة والإشراف

٢ - شارع قشاطر شبرا القاهرة

٧٧٣٢٢١٠٥

هذا الكتاب ..

يقدم لك لغة « لوجو » التى تحوّل جهاز الكمبيوتر إلى صديق متحدّث ، يحاورك بلغة مفهومة بعيدة عن الرموز والمصطلحات .

إن لغة « لوجو » هى المدخل المناسب للأطفال للتعرف على الكمبيوتر والتعامل معه وبرمجته . وهى أيضاً لغة المستقبل للصغار والكبار معاً . فهى اللغة الموجهة للتحكم فى الإنسان الآلى (الروبوت) وهى شقيقة لغة ليسب (LISP) لغة الذكاء الصناعى .

إن لغة « لوجو » تستطيع أن تقدم كل ما تقدمه اللغات الأخرى من إمكانيات ، ولكن فريقاً من الباحثين المهرة قد بذلوا الجهد الكثير فى تطويرها حتى تصل إلينا فى هذا الثوب السهل الممتع بعيداً عن تعقيدات الآلة الإلكترونية .

ولغة « لوجو » هى لغة قياسية لم تنتشر فيها اللهجات المختلفة كما فى لغة بيسك . لذلك يسهل تطبيق برامجها على كل أجهزة الكمبيوتر . بل يصل الأمر إلى إمكانية ابتكار أوامر جديدة وإضافتها إلى لغة لوجو لجهازك الخاص .

ومن أهم ما تفرّد به لغة لوجو هى القدرة على أداء الرسومات عالية الدقة بأوامر بسيطة تصدرها إلى السلحفاه البحرية التى تتحرك أمامك على الشاشة . هذا فضلاً عن قدرتها على التلوين وإصدار الأصوات الموسيقية مما يؤهلها لتكون وسيلة فعالة لبرمجة الألعاب الكمبيوترية .